```
LLL                  IIIIIIIIII     BBBBBBBBBBBBB     RRRRRRRRRRRRR        AAAAAAAAA      RRRRRRRRRRRRR
LLL                  IIIIIIIIII     BBBBBBBBBBBBB     RRRRRRRRRRRRR        AAAAAAAAA      RRRRRRRRRRRRR
LLL                  IIIIIIIIII     BBBBBBBBBBBBB     RRRRRRRRRRRRR        AAAAAAAAA      RRRRRRRRRRRRR
LLL                      III        BBB          BBB  RRR          RRR  AAA          AAA  RRR          RRR
LLL                      III        BBB          BBB  RRR          RRR  AAA          AAA  RRR          RRR
LLL                      III        BBB          BBB  RRR          RRR  AAA          AAA  RRR          RRR
LLL                      III        BBB          BBB  RRR          RRR  AAA          AAA  RRR          RRR
LLL                      III        BBB          BBB  RRR          RRR  AAA          AAA  RRR          RRR
LLL                      III        BBB          BBB  RRR          RRR  AAA          AAA  RRR          RRR
LLL                      III        BBBBBBBBBBBBB     RRRRRRRRRRRRR     AAA          AAA  RRRRRRRRRRRRR
LLL                      III        BBBBBBBBBBBBB     RRRRRRRRRRRRR     AAA          AAA  RRRRRRRRRRRRR
LLL                      III        BBBBBBBBBBBBB     RRRRRRRRRRRRR     AAA          AAA  RRRRRRRRRRRRR
LLL                      III        BBB          BBB  RRR    RRR        AAAAAAAAAAAAAAAA  RRR    RRR
LLL                      III        BBB          BBB  RRR    RRR        AAAAAAAAAAAAAAAA  RRR    RRR
LLL                      III        BBB          BBB  RRR    RRR        AAAAAAAAAAAAAAAA  RRR    RRR
LLL                      III        BBB          BBB  RRR       RRR     AAA          AAA  RRR       RRR
LLL                      III        BBB          BBB  RRR       RRR     AAA          AAA  RRR       RRR
LLL                      III        BBB          BBB  RRR       RRR     AAA          AAA  RRR       RRR
LLLLLLLLLLLLLLL      IIIIIIIIII     BBBBBBBBBBBBB     RRR          RRR  AAA          AAA  RRR          RRR
LLLLLLLLLLLLLLL      IIIIIIIIII     BBBBBBBBBBBBB     RRR          RRR  AAA          AAA  RRR          RRR
LLLLLLLLLLLLLLL      IIIIIIIIII     BBBBBBBBBBBBB     RRR          RRR  AAA          AAA  RRR          RRR
```

FILEID**GETCMD

```
  GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT    CCCCCCCC  MM      MM  DDDDDDDD
  GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT    CCCCCCCC  MM      MM  DDDDDDDD
GG          EE              TT      CC          MMMM  MMMM  DD      DD
GG          EE              TT      CC          MMMM  MMMM  DD      DD
GG          EE              TT      CC          MM  MM  MM  DD      DD
GG          EE              TT      CC          MM  MM  MM  DD      DD
GG          EEEEEEEE        TT      CC          MM      MM  DD      DD
GG          EEEEEEEE        TT      CC          MM      MM  DD      DD
GG   GGGGGG EE              TT      CC          MM      MM  DD      DD
GG   GGGGGG EE              TT      CC          MM      MM  DD      DD
GG      GG  EE              TT      CC          MM      MM  DD      DD      ....
GG      GG  EE              TT      CC          MM      MM  DD      DD      ....
  GGGGGG    EEEEEEEEEE      TT        CCCCCCCC  MM      MM  DDDDDDDD        ....
  GGGGGG    EEEEEEEEEE      TT        CCCCCCCC  MM      MM  DDDDDDDD        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE LIB_GETCMD (                          ! Get LIBRARIAN command line
;     2          0002  0                 LANGUAGE (BLISS32),
;     3          0003  0                 IDENT = 'V04-000'
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1
;     7          0007  1 !
;     8          0008  1 !**********************************************************************
;     9          0009  1 !*                                                                    *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                           *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.            *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                             *
;    13          0013  1 !*                                                                    *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;    19          0019  1 !*  TRANSFERRED.                                                       *
;    20          0020  1 !*                                                                    *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;    23          0023  1 !*  CORPORATION.                                                       *
;    24          0024  1 !*                                                                    *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
;    27          0027  1 !*                                                                    *
;    28          0028  1 !*                                                                    *
;    29          0029  1 !**********************************************************************
;    30          0030  1
;    31          0031  1 !++
;    32          0032  1 !
;    33          0033  1 ! FACILITY:  Library command processor
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !
;    37          0037  1 !       The VAX/VMS librarian is invoked by DCL to process the LIBRARY
;    38          0038  1 !       command.  It utilizes the librarian procedure set to perform
;    39          0039  1 !       the actual modifications to the library.
;    40          0040  1 !
;    41          0041  1 ! ENVIRONMENT:
;    42          0042  1 !
;    43          0043  1 !       VAX native, user mode.
;    44          0044  1 !
;    45          0045  1 !--
;    46          0046  1 !
;    47          0047  1 !
;    48          0048  1 ! AUTHOR:  Benn Schreiber,       CREATION DATE:  11-June-1979
;    49          0049  1 !
;    50          0050  1 ! MODIFIED BY:
;    51          0051  1 !
;    52          0052  1 !       V03-007 GJA0085         Greg Awdziewicz         27-Jun-1984
;    53          0053  1 !               - Don't overwrite lib$gl_type and default extensions
;    54          0054  1 !               which were set because of a library-type qualifier just
;    55          0055  1 !               because an extension was specified on the library file
;    56          0056  1 !               specification.  (Important especially for inserting into
;    57          0057  1 !               a shareable image library.)
```

```
;    58          0058  1 ;
;    59          0059  1 ;       V03-006 GJA0075         Greg Awdziewicz         10-Feb-1984
;    60          0060  1 ;               - Use the new improved LIB$File_scan with multiple
;    61          0061  1 ;               sticky file and searchlist capability.  Remove uses
;    62          0062  1 ;               of related filename nameblock.
;    63          0063  1 ;
;    64          0064  1 ;       V03-005 MCN0145         Maria del C. Nasr       07-Feb-1984
;    65          0065  1 ;               Add qualifier /DATA with keywords REDUCE and EXPAND
;    66          0066  1 ;               to replace /COMPRESS=REDUCE.  We will share the same
;    67          0067  1 ;               flags, and routines originally used with /COMPRESS.
;    68          0068  1 ;
;    69          0069  1 ;       V03-004 JWT0115         Jim Teague              27-Apr-1983
;    70          0070  1 ;               Made /SQUEEZE default.
;    71          0071  1 ;
;    72          0072  1 ;       V03-003 JWT0103         Jim Teague              05-Apr-1983
;    73          0073  1 ;               New CLI interface.
;    74          0074  1 ;
;    75          0075  1 ;       V03-002 JWT0056         Jim Teague              16-Sep-1982
;    76          0076  1 ;               Implemented DCX interface for /COMPRESS=REDUCE
;    77          0077  1 ;
;    78          0078  1 ;       V03-001         RPG0301         Bob Grosso      27-Apr-1982
;    79          0079  1 ;               Add subtitles to listing and fix /QUAL="" so the
;    80          0080  1 ;               novalue error message has the name of the qualifier
;    81          0081  1 ;               filled in.
;    82          0082  1 ;
;    83          0083  1 ;--
```

```
;    85          0084  1 %SBTTL 'Declarations';
;    86          0085  1
;    87          0086  1 LIBRARY
;    88          0087  1         'SYS$LIBRARY:STARLET.L32';              !System data structure definitions
;    89          0088  1 LIBRARY
;    90          0089  1         'SYS$LIBRARY:CLIMAC.L32';               !Command language data structure defs.
;    91          0090  1 LIBRARY
;    92          0091  1         'SYS$LIBRARY:TPAMAC.L32';               !Table driven finite state parser macros
;    93          0092  1 REQUIRE
;    94          0093  1         'PREFIX';                               !Macro defs, etc.
;    95          0277  1 REQUIRE
;    96          0278  1         'LIBDEF';                               !Librarian command interface definitions
;    97          0566  1
;    98        P 0567  1 sd ( 'MODULES', 'BEFORE', 'COMPRESS', 'CREATE', 'CROSS_REFERENCE', 'DATA',
;    99        P 0568  1      'DELETE', 'EXTRACT', 'FULL', 'GLOBALS', 'HELP', 'HISTORY',
;   100        P 0569  1      'INSERT', 'LIST', 'LOG', 'MACRO', 'NAMES',
;   101        P 0570  1      'OBJECT', 'ONLY', 'OUTPUT', 'REMOVE', 'REPLACE', 'SELECTIVE_SEARCH', 'SHARE',
;   102          0571  1      'SINCE', 'SQUEEZE', 'TEXT', 'WIDTH', 'P1', 'P2' );
;   103          0572  1
;   104          0573  1 LITERAL
;   105          0574  1     qualtabsiz = 24,
;   106          0575  1     fdbsize = fdb$c_size + nam$c_bln            !Define total size of an FDB
;   107          0576  1                 + (2 * nam$c_maxrss);
;   108          0577  1
;   109          0578  1 EXTERNAL ROUTINE
;   110          0579  1     lib$cvt_dtb : addressing_mode(general),
;   111          0580  1     cli$get_value,
;   112          0581  1     cli$present,
;   113          0582  1     getfilnamdesc,                              !Return string descriptor for filename
;   114          0583  1     lib_get_mem,                                !Allocate virtual memory
;   115          0584  1     lib$cvt_time : ADDRESSING_MODE (GENERAL),   !convert ascii time to binary
;   116          0585  1     lib$file_scan : ADDRESSING_MODE (GENERAL),  !Find wild-card files
;   117          0586  1     lib$tparse : ADDRESSING_MODE (GENERAL),     !Table driven parser
;   118          0587  1     lib$lookup_key : ADDRESSING_MODE (GENERAL), !Look up qualifier keyword
;   119          0588  1     lib_free_mem;                               !Deallocate virtual memory
;   120          0589  1
;   121          0590  1 FORWARD ROUTINE
;   122          0591  1     clearsqueeze,                               !Clear local squeeze bit
;   123          0592  1     setsqueeze,                                 !Set local squeeze bit
;   124          0593  1     selectcrosref,                              !Select cross reference options
;   125          0594  1     scan_options,                               !Scan create/compress/data options
;   126          0595  1     call_tparse,                                !call lib$tparse
;   127          0596  1     syntaxerr,                                  !report tparse syntaxerr
;   128          0597  1     value_req,                                  !Set flag that a value is required on create/compress option
;   129          0598  1     set_macro_type,                             !/MACRO
;   130          0599  1     set_object_type,                            !/OBJECT
;   131          0600  1     set_help_type,                              !/HELP
;   132          0601  1     set_text_type,                              !/TEXT
;   133          0602  1     set_shr_type,                               !/SHARE
;   134          0603  1     set_lib_type,                               !Does the work for set_xx_type
;   135          0604  1     listonlymods,                               !/ONLY
;   136          0605  1     setmodulename,                              !/MODULES
;   137          0606  1     compressfile,                               !/COMPRESS
;   138          0607  1     createlib,                                  !/CREATE
;   139          0608  1     datareduce,                                 !/DATA
;   140          0609  1     extractfile,                                !/EXTRACT
;   141          0610  1     deletemodules,                              !/DELETE
```

```
;  142          0611  1      setlistfile,                    !/LIST
;  143          0612  1      setlistwidth,                   !/WIDTH
;  144          0613  1      removesymbols,                  !/REMOVE
;  145          0614  1      outputfile,                     !/OUTPUT processor
;  146          0615  1      input1file,                     !Primary input file processor
;  147          0616  1      input2file,                     !Secondary input file processor
;  148          0617  1      get_name_list,                  !Extract args from qualifier
;  149          0618  1      filescanfruact,                 !Action routine for successful file scan
;  150          0619  1      filescanflsact,                 !Action routine for unsuccessful scan
;  151          0620  1      initialize_fdb,                 !Initialize an FDB
;  152          0621  1      date_parse,                     !Parse date for /BEFORE and /SINCE
;  153          0622  1      since_date,
;  154          0623  1      before_date,
;  155          0624  1      allocate_fdb,                   !Allocate file descriptor block
;  156          0625  1      clierror : novalue;             !Process CLI errors
;  157          0626  1
;  158          0627  1  EXTERNAL
;  159          0628  1      lib$before_date : BBLOCK,       !Date from /BEFORE
;  160          0629  1      lib$since_date : BBLOCK,        !Date from /SINCE
;  161          0630  1      lib$al_creopts : VECTOR [,LONG], !Array of create option values
;  162          0631  1      lib$gl_cre8flags : BITVECTOR,   !Bit flags for specified create/compress options
;  163          0632  1      lib$gl_listwid,                 !width of library list line
;  164          0633  1      lib$gl_type,                    !Library type
;  165          0634  1      lib$gl_modlisl : VECTOR [2],    !Listhead for "list only" name list
;  166          0635  1      lib$gl_modxtrl : VECTOR [2],    !Listhead for module extraction
;  167          0636  1      lib$gl_modupdl : VECTOR [2],    !Listhead for library updates
;  168          0637  1      lib$gl_objsyrml : VECTOR [2],   !Listhead for object module symbol removal
;  169          0638  1      lib$gl_delmodl : VECTOR [2],    !Listhead for deleted module list
;  170          0639  1      lib$gl_ctlmsk : BLOCK [2],      !Control mask for Librarian
;  171          0640  1      lib$gl_libfdb : REF BBLOCK,     !FDB for library file
;  172          0641  1      lib$gl_outfdb : REF BBLOCK,     !FDB for output file
;  173          0642  1      lib$gl_lisfdb : REF BBLOCK,     !FDB for listing file
;  174          0643  1      lib$gl_tmpfdb : REF BBLOCK,     !FDB for temporary use
;  175          0644  1      lib$gl_inplist : REF BBLOCK,    !Listhead for input files list
;  176          0645  1      lib_mac_defext : BBLOCK,        !Default extension for macro source files
;  177          0646  1      lib_obj_defext : BBLOCK,        !...for object files
;  178          0647  1      lib_hlp_defext : BBLOCK,        !...for help files
;  179          0648  1      lib_txt_defext : BBLOCK,        !...for text files
;  180          0649  1      lib_shr_defext : BBLOCK;        !...for shr img stb libraries
;  181          0650  1
;  182          0651  1  GLOBAL
;  183          0652  1      lib$gl_tpindex,
;  184          0653  1      lib$gl_valreq,
;  185          0654  1      lib$gl_creflags : BLOCK [1];    !Cross reference flags
;  186          0655  1
;  187          0656  1  OWN
;  188          0657  1
;  189          0658  1      qual_table : vector [ qualtabsiz, long ] initial
;  190          0659  1                 ( sd_before, sd_compress, sd_create, sd_cross_reference,
;  191          0660  1                   sd_data, sd_delete, sd_extract, sd_full,
;  192          0661  1                   sd_globals, sd_help, sd_history, sd_list, sd_log, sd_macro, sd_names, sd_object,
;  193          0662  1                   sd_only, sd_remove, sd_share, sd_since, sd_text, sd_replace, sd_insert, sd_squeeze ),
;  194          0663  1
;  195          0664  1      bit_table : vector [ qualtabsiz, byte ] initial ( byte
;  196          0665  1                 ( $bitposition(lib$v_before),$bitposition(lib$v_compress),
;  197          0666  1                   $bitposition(lib$v_create),$bitposition(lib$v_cross),
;  198          0667  1                   $bitposition(lib$v_data),$bitposition(lib$v_delete),$bitposition(lib$v_extract),
```

```
;  199          0668  1                    $bitposition(lib$v_full),$bitposition(lib$v_globals),$bitposition(lib$v_help),
;  200          0669  1                    $bitposition(lib$v_history),$bitposition(lib$v_list),$bitposition(lib$v_log),
;  201          0670  1                    $bitposition(lib$v_macro),$bitposition(lib$v_names),$bitposition(lib$v_object),
;  202          0671  1                    $bitposition(lib$v_only),$bitposition(lib$v_remove),$bitposition(lib$v_shrstb),
;  203          0672  1                    $bitposition(lib$v_since),$bitposition(lib$v_text),$bitposition(lib$v_replace),
;  204          0673  1                    $bitposition(lib$v_insert),$bitposition(lib$v_squeeze)   )),
;  205          0674  1
;  206          0675  1      call_table : vector [ qualtabsiz, long ] initial
;  207          0676  1                   ( before_date, compressfile, createlib, selectcrosref,
;  208          0677  1                     datareduce, deletemodules, extractfile, 0, 0,
;  209          0678  1                     set_help_type, 0, setlistfile, 0, set_macro_type, 0, set_object_type, listonlymods,
;  210          0679  1                     removesymbols, set_shr_type, since_date, set_text_type, 0, 0, 0 ),
;  211          0680  1
;  212        P 0681  1      crf_table : $lib_key_table (
;  213          0682  1                   ( SYMBOL, 0 ), ( VALUE, 1 ), ( MODULE, 2 ) ),
;  214        P 0683  1      opt_table : $lib_key_table (
;  215          0684  1                   ( ALL, 0 ), ( NONE, 1 ) ),
;  216          0685  1 !
;  217          0686  1 ! Other OWN storage
;  218          0687  1 !
;  219          0688  1      list_desc : dynamic_descriptor,
;  220          0689  1      squeeze_flag : INITIAL (1),                    !Cleared if /NOSQUEEZE.
;  221          0690  1      def_lib_extn : BBLOCK [dsc$c_s_bln],           !Default extension for libraries
;  222          0691  1      def_fil_extn : BBLOCK [dsc$c_s_bln],           !Default extension for other files
;  223          0692  1      lastfdb : REF BBLOCK,                          !Pointer to last FDB allocated
;  224          0693  1      prevfdb : REF BBLOCK,                          !Pointer to previous of last
;  225          0694  1      sysoutputdesc : descriptor ('SYS$OUTPUT:'),    !String descriptor for SYS$OUTPUT
;  226          0695  1      modulenamedesc : dynamic_descriptor,           !String descriptor for /MODULES=
;  227          0696  1      cliworkptr,                                    !Pointer to cli work area
;  228          0697  1      filesnotfound,                                 !Count of not found files
;  229          0698  1      token_desc : dynamic_descriptor,
;  230          0699  1      crf_bysymbol : descriptor ('SYMBOL'),          !Cross reference by symbol
;  231          0700  1      crf_byvalue  : descriptor ('VALUE'),           !Cross reference by value
;  232          0701  1      crf_bymodule : descriptor ('MODULE'),          !Cross reference by module
;  233          0702  1      crf_bynone : LONG INITIAL (0),                 !End of crf options
;  234          0703  1      all_options : descriptor ('ALL'),              !All of the options
;  235          0704  1      no_options : descriptor ('NONE'),              !None of the options
;  236          0705  1      end_options : LONG INITIAL (0),                !End of all/none
;  237          0706  1
;  238          0707  1      tpa_block : BBLOCK [tpa$k_length0],            !tparse block
;  239          0708  1      tpa_desc : REF BBLOCK;                         !pointer to current string descriptor
;  240          0709  1
```

```
;  242          0710  1 %SBTTL 'TPARSE tables';
;  243          0711  1
;  244          0712  1 !
;  245          0713  1 !   TPARSE tables to parse /CREATE=() qualifier values.
;  246          0714  1 !
;  247          0715  1 $INIT_STATE (create_states, create_keys);
;  248          0716  1
;  249        P 0717  1 $STATE (,
;  250        P 0718  1         ('BLOCKS',tpa$_exit,value_req,lib$c_opt_blks, lib$gl_tpindex,true),
;  251        P 0719  1         ('GLOBALS',tpa$_exit,value_req,lib$c_opt_gbls, lib$gl_tpindex,true),
;  252        P 0720  1         ('MODULES',tpa$_exit,value_req,lib$c_opt_mods, lib$gl_tpindex,true),
;  253        P 0721  1         ('KEYSIZE',tpa$_exit,value_req,lib$c_opt_ksz, lib$gl_tpindex,true),
;  254        P 0722  1         ('HISTORY',tpa$_exit,value_req,lib$c_opt_luhs, lib$gl_tpindex,true),
;  255        P 0723  1         ('VERSION',tpa$_exit,value_req,lib$c_opt_ver, lib$gl_tpindex,true),
;  256        P 0724  1         (tpa$_lambda,tpa$_exit,syntaxerr, , ,lib$_badkey),
;  257          0725  1         );
;  258          0726  1
;  259          0727  1
;  260          0728  1 $INIT_STATE (compress_states, compress_keys);
;  261          0729  1
;  262        P 0730  1 $STATE (,
;  263        P 0731  1         ('BLOCKS',tpa$_exit,value_req,lib$c_opt_blks, lib$gl_tpindex,true),
;  264        P 0732  1         ('GLOBALS',tpa$_exit,value_req,lib$c_opt_gbls, lib$gl_tpindex,true),
;  265        P 0733  1         ('MODULES',tpa$_exit,value_req,lib$c_opt_mods, lib$gl_tpindex,true),
;  266        P 0734  1         ('KEYSIZE',tpa$_exit,value_req,lib$c_opt_ksz, lib$gl_tpindex,true),
;  267        P 0735  1         ('HISTORY',tpa$_exit,value_req,lib$c_opt_luhs, lib$gl_tpindex,true),
;  268        P 0736  1         ('VERSION',tpa$_exit,value_req,lib$c_opt_ver, lib$gl_tpindex,true),
;  269        P 0737  1         ('KEEP', tpa$_exit,value_req,1^lib$c_opt_keep, lib$gl_cre8flags,false),
;  270        P 0738  1         ('REDUCE', tpa$_exit,value_req,1^lib$c_opt_dcx, lib$gl_cre8flags,false),
;  271        P 0739  1         (tpa$_lambda,tpa$_exit,syntaxerr,,, lib$_badkey),
;  272          0740  1         );
;  273          0741  1 !
;  274          0742  1 ! TPARSE tables for /DATA qualifier options
;  275          0743  1 !
;  276          0744  1 $INIT_STATE (data_states, data_keys);
;  277          0745  1
;  278        P 0746  1 $STATE (,
;  279        P 0747  1         ('REDUCE', tpa$_exit,value_req,1^lib$c_opt_dcx, lib$gl_cre8flags,false),
;  280        P 0748  1         ('EXPAND', tpa$_exit,value_req,,,false),
;  281        P 0749  1         (tpa$_lambda,tpa$_exit,syntaxerr,,, lib$_badkey),
;  282          0750  1         );
```

```
;  284     0751  1 %SBTTL 'LIB_GET_COMMAND';
;  285     0752  1
;  286     0753  1 GLOBAL ROUTINE lib_get_command  (arglist) =
;  287     0754  2 BEGIN
;  288     0755  2
;  289     0756  2 !++
;  290     0757  2 ! This routine initializes the CLI result parser and then re-calls it for the
;  291     0758  2 ! processing of each command.
;  292     0759  2 !
;  293     0760  2 ! Inputs:
;  294     0761  2 !
;  295     0762  2 !       arglist         pointer to the argument list that invoked the image
;  296     0763  2 !
;  297     0764  2 ! Outputs:
;  298     0765  2 !
;  299     0766  2 !       the command is parsed, data structures filled in.
;  300     0767  2 !
;  301     0768  2 !--
;  302     0769  2 MAP
;  303     0770  2         arglist : REF BBLOCK;
;  304     0771  2 LOCAL
;  305     0772  2         status;
;  306     0773  2
;  307     0774  2 lastfdb = lib$gl_inplist;                       !Init the last FDB pointer
;  308     0775  2 prevfdb = lib$gl_inplist;                       ! and the prev. fdb pointer
;  309     0776  2 lib$gl_inplist = 0;                             !Init input FDB listhead
;  310     0777  2
;  311     0778  2 ! Initialize the library type as "unknown" but set the default file
;  312     0779  2 ! extensions as if the library type is "object", which will be the
;  313     0780  2 ! default if no other mechanism changes it:
;  314     0781  2 !
;  315     0782  2 set_lib_type (lbr$c_typ_unk, lib_obj_defext);
;  316     0783  2
;  317     0784  2 !
;  318     0785  2 ! Initialize the name list queue headers
;  319     0786  2 !
;  320     0787  2 lib$gl_modlisl [0] = lib$gl_modlisl;
;  321     0788  2 lib$gl_modlisl [1] = lib$gl_modlisl;
;  322     0789  2 lib$gl_modxtrl [0] = lib$gl_modxtrl;
;  323     0790  2 lib$gl_modxtrl [1] = lib$gl_modxtrl;
;  324     0791  2 lib$gl_modupdl [0] = lib$gl_modupdl;
;  325     0792  2 lib$gl_modupdl [1] = lib$gl_modupdl;
;  326     0793  2 lib$gl_objsyrml [0] = lib$gl_objsyrml;
;  327     0794  2 lib$gl_objsyrml [1] = lib$gl_objsyrml;
;  328     0795  2 lib$gl_delmodl [0] = lib$gl_delmodl;
;  329     0796  2 lib$gl_delmodl [1] = lib$gl_delmodl;
;  330     0797  2
;  331     0798  2 !
;  332     0799  2 ! Loop through most qualifiers, setting the appropriate bit
;  333     0800  2 !
;  334     0801  2 incr i to qualtabsiz - 1 do
;  335     0802  3     if (lib$gl_ctlmsk[0,.bit_table[.i],1,0] = cli$present( .qual_table[.i] ))
;  336     0803  2     then
;  337     0804  2         !
;  338     0805  2         ! If an address of a routine is supplied, call it.
;  339     0806  2         !
;  340     0807  2         if .call_table[.i] neq 0
```

```
;  341     0808  2          then
;  342     0809  2              (.call_table[.i])();
;  343     0810  2
;  344     0811  2 if .lib$gl_ctlmsk[lib$v_squeeze]
;  345     0812  2 then
;  346     0813  2     setsqueeze()
;  347     0814  2 else
;  348     0815  2     clearsqueeze();
;  349     0816  2
;  350     0817  2 if cli$present(sd_width)
;  351     0818  2 then
;  352     0819  2     setlistwidth();
;  353     0820  2
;  354     0821  2 !
;  355     0822  2 ! Get library file name
;  356     0823  2 !
;  357     0824  2 cli$get_value(sd_p1, token_desc);
;  358     0825  2 inputlfile();
;  359     0826  2
;  360     0827  2 !
;  361     0828  2 ! Get output file name
;  362     0829  2 !
;  363     0830  2 cli$get_value(sd_output,token_desc);
;  364     0831  2 outputfile();
;  365     0832  2
;  366     0833  2 filesnotfound = 0;
;  367     0834  2
;  368     0835  3 IF NOT (.lib$gl_ctlmsk [lib$v_compress]                    !Unless compress
;  369     0836  3         OR .lib$gl_ctlmsk [lib$v_data]                     ! or data
;  370     0837  3         OR .lib$gl_ctlmsk [lib$v_extract]                  ! or extract
;  371     0838  3         OR .lib$gl_ctlmsk [lib$v_cross] )                  !           or cross ref
;  372     0839  2 THEN
;  373     0840  2     WHILE cli$get_value(sd_p2, token_desc) DO
;  374     0841  3     BEGIN
;  375     0842  3     lib$gl_ctlmsk [lib$v_selective] = false;               !Clear selective bit
;  376     0843  3     if cli$present(sd_modules)
;  377     0844  3     then
;  378     0845  3         setmodulename();
;  379     0846  3     input2file(token_desc);
;  380     0847  3     IF .lib$gl_ctlmsk [lib$v_selective]                    !If selective search was seen
;  381     0848  3     THEN lastfdb [fdb$v_setsel] = true;
;  382     0849  2     END;
;  383     0850  2
;  384     0851  2 IF .lib$gl_inplist EQL 0                                   !If all files specified
;  385     0852  2     AND .filesnotfound NEQ 0                               ! were not found
;  386     0853  2     THEN IF .lib$gl_ctlmsk [lib$v_create]                  !If creating
;  387     0854  2             THEN RETURN false                              ! then nothing to do
;  388     0855  2             ELSE IF NOT .lib$gl_ctlmsk [lib$v_delete]      ! otherwise, if no other function selected
;  389     0856  2                         AND NOT .lib$gl_ctlmsk [lib$v_extract]
;  390     0857  2                         AND NOT .lib$gl_ctlmsk [lib$v_list]
;  391     0858  2                         AND NOT .lib$gl_ctlmsk [lib$v_remove]
;  392     0859  2                         AND NOT .lib$gl_ctlmsk [lib$v_data]
;  393     0860  2                         AND NOT .lib$gl_ctlmsk [lib$v_compress]
;  394     0861  2                         AND NOT .lib$gl_ctlmsk [lib$v_cross]
;  395     0862  2                 THEN RETURN false;                         !  then nothing to do
;  396     0863  2
;  397     0864  2 IF .lib$gl_inplist NEQ 0                                   !If input files are present
```

```
;  398          0865  2      AND NOT .lib$gl_ctlmsk [lib$v_insert]                    ! and did not say insert or replace
;  399          0866  2  THEN lib$gl_ctlmsk [lib$v_replace] = lib$gl_ctlmsk [lib$v_insert] = true;       ! then make sure this is a r
;  400          0867  2  !
;  401          0868  2  ! Terminate parse
;  402          0869  2  !
;  403          0870  2  CH$MOVE (dsc$c_s_bln, def_lib_extn, lib$gl_libfdb [fdb$l_defext]);
;  404          0871  2  IF .lib$gl_ctlmsk [lib$v_insert] OR .lib$gl_ctlmsk [lib$v_delete] !If insert, remove
;  405          0872  2  OR .lib$gl_ctlmsk [lib$v_remove]                               !or delete
;  406          0873  2  THEN allocate_fdb (lib$gl_tmpfdb);                            !Allocate FDB in case
;  407          0874  2                                                                !old format library
;  408          0875  2  IF .lib$gl_creflags EQL 0                                     !If /CROSS=NONE
;  409          0876  2  THEN lib$gl_ctlmsk [lib$v_cross] = false;                      ! then disable it
;  410          0877  2  RETURN true
;  411          0878  1  END;


                                                .TITLE  LIB_GETCMD
                                                .IDENT  \V04-000\

                                                .PSECT  _LIB$KEY1$,NOWRT,  SHR,  PIC,1

                                        00000 ;TPA$KEYSTO
                                        U.2:    .BLKB   0
                          53  4B  43  4F  4C  42  00000 ;TPA$KEYST
                                        U.4:    .ASCII  \BLOCKS\                        ;
                                      FF  00006         .BYTE   -1                      ;
                                        00007 ;TPA$KEYSTO
                                        U.12:   .BLKB   0
                      53  4C  41  42  4F  4C  47  00007 ;TPA$KEYST
                                        U.14:   .ASCII  \GLOBALS\                       ;
                                      FF  0000E         .BYTE   -1                      ;
                                        0000F ;TPA$KEYSTO
                                        U.22:   .BLKB   0
                      53  45  4C  55  44  4F  4D  0000F ;TPA$KEYST
                                        U.24:   .ASCII  \MODULES\                       ;
                                      FF  00016         .BYTE   -1                      ;
                                        00017 ;TPA$KEYSTO
                                        U.32:   .BLKB   0
                      45  5A  49  53  59  45  4B  00017 ;TPA$KEYST
                                        U.34:   .ASCII  \KEYSIZE\                       ;
                                      FF  0001E         .BYTE   -1                      ;
                                        0001F ;TPA$KEYSTO
                                        U.42:   .BLKB   0
                      59  52  4F  54  53  49  48  0001F ;TPA$KEYST
                                        U.44:   .ASCII  \HISTORY\                       ;
                                      FF  00026         .BYTE   -1                      ;
                                        00027 ;TPA$KEYSTO
                                        U.52:   .BLKB   0
                      4E  4F  49  53  52  45  56  00027 ;TPA$KEYST
                                        U.54:   .ASCII  \VERSION\                       ;
                                      FF  0002E         .BYTE   -1                      ;
                                      FF  0002F ;TPA$KEYFILL
                                        U.67:   .BYTE   -1                              ;
                                        00030 ;TPA$KEYSTO
                                        U.69:   .BLKB   0
                          53  4B  43  4F  4C  42  00030 ;TPA$KEYST
                                        U.71:   .ASCII  \BLOCKS\                        ;
```

```
                                   FF  00036           .BYTE   -1                                        ;
                                       00037 ;TPA$KEYST0
                                             U.79:     .BLKB   0
      53  4C  41  42  4F  4C  47  00037 ;TPA$KEYST
                                             U.81:     .ASCII  \GLOBALS\                                 ;
                                   FF  0003E           .BYTE   -1                                        ;
                                       0003F ;TPA$KEYST0
                                             U.89:     .BLKB   0
      53  45  4C  55  44  4F  4D  0003F ;TPA$KEYST
                                             U.91:     .ASCII  \MODULES\                                 ;
                                   FF  00046           .BYTE   -1                                        ;
                                       00047 ;TPA$KEYST0
                                             U.99:     .BLKB   0
      45  5A  49  53  59  45  4B  00047 ;TPA$KEYST
                                             U.101:    .ASCII  \KEYSIZE\                                 ;
                                   FF  0004E           .BYTE   -1                                        ;
                                       0004F ;TPA$KEYST0
                                             U.109:    .BLKB   0
      59  52  4F  54  53  49  48  0004F ;TPA$KEYST
                                             U.111:    .ASCII  \HISTORY\                                 ;
                                   FF  00056           .BYTE   -1                                        ;
                                       00057 ;TPA$KEYST0
                                             U.119:    .BLKB   0
      4E  4F  49  53  52  45  56  00057 ;TPA$KEYST
                                             U.121:    .ASCII  \VERSION\                                 ;
                                   FF  0005E           .BYTE   -1                                        ;
                                       0005F ;TPA$KEYST0
                                             U.129:    .BLKB   0
                      50  45  45  4B  0005F ;TPA$KEYST
                                             U.131:    .ASCII  \KEEP\                                    ;
                                   FF  00063           .BYTE   -1                                        ;
                                       00064 ;TPA$KEYST0
                                             U.139:    .BLKB   0
          45  43  55  44  45  52  00064 ;TPA$KEYST
                                             U.141:    .ASCII  \REDUCE\                                  ;
                                   FF  0006A           .BYTE   -1                                        ;
                                   FF  0006B ;TPA$KEYFILL
                                             U.154:    .BYTE   -1                                        ;
                                       0006C ;TPA$KEYST0
                                             U.156:    .BLKB   0
          45  43  55  44  45  52  0006C ;TPA$KEYST
                                             U.158:    .ASCII  \REDUCE\                                  ;
                                   FF  00072           .BYTE   -1                                        ;
                                       00073 ;TPA$KEYST0
                                             U.166:    .BLKB   0
          44  4E  41  50  58  45  00073 ;TPA$KEYST
                                             U.168:    .ASCII  \EXPAND\                                  ;
                                   FF  00079           .BYTE   -1                                        ;
                                   FF  0007A ;TPA$KEYFILL
                                             U.179:    .BYTE   -1                                        ;

                                                       .PSECT  _LIB$STATES,NOWRT,  SHR,  PIC,1

                                       00000 CREATE_STATES::
                                                       .BLKB   0
                                 F300  00000 ;TPA$TYPE
                                             U.5:      .WORD   -3328                                     ;
```

LIB
V04

; R

```
                     01  00002 ;TPA$FLAGS2
                         U.6:    .BYTE   1                                       ;
00000001             00003 ;TPA$PARAM
                         U.7:    .LONG   1                                       ;
00000000V            00007 ;TPA$ACTION
                         U.8:    .LONG   <<VALUE_REQ-U.8>-4>                     ;
00000000*            0000B ;TPA$ADDR
                         U.9:    .LONG   <<LIB$GL_TPINDEX-U.9>-4>                ;
00000000             0000F ;TPA$MASK
                         U.10:   .LONG   0                                       ;
                   FFFF  00013 ;TPA$TARGET
                         U.11:   .WORD   -1                                      ;
                   F301  00015 ;TPA$TYPE
                         U.15    .WORD   -3327                                   ;
                     01  00017 ;TPA$FLAGS2
                         U.16:   .BYTE   1                                       ;
00000001             00018 ;TPA$PARAM
                         U.17:   .LONG   1                                       ;
00000000V            0001C ;TPA$ACTION
                         U.18:   .LONG   <<VALUE_REQ-U.18>-4>                    ;
00000000*            00020 ;TPA$ADDR
                         U.19:   .LONG   <<LIB$GL_TPINDEX-U.19>-4>               ;
00000001             00024 ;TPA$MASK
                         U.20:   .LONG   1                                       ;
                   FFFF  00028 ;TPA$TARGET
                         U.21:   .WORD   -1                                      ;
                   F302  0002A ;TPA$TYPE
                         U.25:   .WORD   -3326                                   ;
                     01  0002C ;TPA$FLAGS2
                         U.26:   .BYTE   1                                       ;
00000001             0002D ;TPA$PARAM
                         U.27:   .LONG   1                                       ;
00000000V            00031 ;TPA$ACTION
                         U.28:   .LONG   <<VALUE_REQ-U.28>-4>                    ;
00000000*            00035 ;TPA$ADDR
                         U.29:   .LONG   <<LIB$GL_TPINDEX-U.29>-4>               ;
00000002             00039 ;TPA$MASK
                         U.30:   .LONG   2                                       ;
                   FFFF  0003D ;TPA$TARGET
                         U.31:   .WORD   -1                                      ;
                   F303  0003F ;TPA$TYPE
                         U.35:   .WORD   -3325                                   ;
                     01  00041 ;TPA$FLAGS2
                         U.36:   .BYTE   1                                       ;
00000001             00042 ;TPA$PARAM
                         U.37:   .LONG   1                                       ;
00000000V            00046 ;TPA$ACTION
                         U.38:   .LONG   <<VALUE_REQ-U.38>-4>                    ;
00000000*            0004A ;TPA$ADDR
                         U.39:   .LONG   <<LIB$GL_TPINDEX-U.39>-4>               ;
00000003             0004E ;TPA$MASK
                         U.40:   .LONG   3                                       ;
                   FFFF  00052 ;TPA$TARGET
                         U.41:   .WORD   -1                                      ;
                   F304  00054 ;TPA$TYPE
                         U.45:   .WORD   -3324                                   ;
                     01  00056 ;TPA$FLAGS2
```

```
                        U.46:   .BYTE   1                               ;
00000001  00057 ;TPA$PARAM
                        U.47:   .LONG   1                               ;
00000000V 0005B ;TPA$ACTION
                        U.48:   .LONG   <<VALUE_REQ-U.48>-4>            ;
00000000* 0005F ;TPA$ADDR
                        U.49:   .LONG   <<LIB$GL_TPINDEX-U.49>-4>       ;
00000004  00063 ;TPA$MASK
                        U.50:   .LONG   4                               ;
    FFFF  00067 ;TPA$TARGET
                        U.51:   .WORD   -1                              ;
    F305  00069 ;TPA$TYPE
                        U.55:   .WORD   -3323                           ;
      01  0006B ;TPA$FLAGS2
                        U.56:   .BYTE   1                               ;
00000001  0006C ;TPA$PARAM
                        U.57:   .LONG   1                               ;
00000000V 00070 ;TPA$ACTION
                        U.58:   .LONG   <<VALUE_REQ-U.58>-4>            ;
00000000* 00074 ;TPA$ADDR
                        U.59:   .LONG   <<LIB$GL_TPINDEX-U.59>-4>       ;
00000005  00078 ;TPA$MASK
                        U.60:   .LONG   5                               ;
    FFFF  0007C ;TPA$TARGET
                        U.61:   .WORD   -1                              ;
    97F6  0007E ;TPA$TYPE
                        U.62:   .WORD   -26634                          ;
      01  00080 ;TPA$FLAGS2
                        U.63:   .BYTE   1                               ;
0086110C  00081 ;TPA$PARAM
                        U.64:   .LONG   8786188                         ;
00000000V 00085 ;TPA$ACTION
                        U.65:   .LONG   <<SYNTAXERR-U.65>-4>            ;
    FFFF  00089 ;TPA$TARGET
                        U.66:   .WORD   -1                              ;
              0008B             .BLKB   1
              0008C COMPRESS_STATES::
                                .BLKB   0
    F300  0008C ;TPA$TYPE
                        U.72:   .WORD   -3328                           ;
      01  0008E ;TPA$FLAGS2
                        U.73:   .BYTE   1                               ;
00000001  0008F ;TPA$PARAM
                        U.74:   .LONG   1                               ;
00000000V 00093 ;TPA$ACTION
                        U.75:   .LONG   <<VALUE_REQ-U.75>-4>            ;
00000000* 00097 ;TPA$ADDR
                        U.76:   .LONG   <<LIB$GL_TPINDEX-U.76>-4>       ;
00000000  0009B ;TPA$MASK
                        U.77:   .LONG   0                               ;
    FFFF  0009F ;TPA$TARGET
                        U.78:   .WORD   -1                              ;
    F301  000A1 ;TPA$TYPE
                        U.82:   .WORD   -3327                           ;
      01  000A3 ;TPA$FLAGS2
                        U.83:   .BYTE   1                               ;
00000001  000A4 ;TPA$PARAM
```

```
                                U.84:   .LONG   1                                       ;
00000000V 000A8                 ;TPA$ACTION
                                U.85:   .LONG   <<VALUE_REQ-U.85>-4>                    ;
00000000* 000AC                 ;TPA$ADDR
                                U.86:   .LONG   <<LIB$GL_TPINDEX-U.86>-4>               ;
00000001  000B0                 ;TPA$MASK
                                U.87:   .LONG   1                                       ;
    FFFF  000B4                 ;TPA$TARGET
                                U.88:   .WORD   -1                                      ;
    F302  000B6                 ;TPA$TYPE
                                U.92:   .WORD   -3326                                   ;
      01  000B8                 ;TPA$FLAGS2
                                U.93:   .BYTE   1                                       ;
00000001  000B9                 ;TPA$PARAM
                                U.94:   .LONG   1                                       ;
00000000V 000BD                 ;TPA$ACTION
                                U.95:   .LONG   <<VALUE_REQ-U.95>-4>                    ;
00000000* 000C1                 ;TPA$ADDR
                                U.96:   .LONG   <<LIB$GL_TPINDEX-U.96>-4>               ;
00000002  000C5                 ;TPA$MASK
                                U.97:   .LONG   2                                       ;
    FFFF  000C9                 ;TPA$TARGET
                                U.98:   .WORD   -1                                      ;
    F303  000CB                 ;TPA$TYPE
                                U.102:  .WORD   -3325                                   ;
      01  000CD                 ;TPA$FLAGS2
                                U.103:  .BYTE   1                                       ;
00000001  000CE                 ;TPA$PARAM
                                U.104:  .LONG   1                                       ;
00000000V 000D2                 ;TPA$ACTION
                                U.105:  .LONG   <<VALUE_REQ-U.105>-4>                   ;
00000000* 000D6                 ;TPA$ADDR
                                U.106:  .LONG   <<LIB$GL_TPINDEX-U.106>-4>              ;
00000003  000DA                 ;TPA$MASK
                                U.107:  .LONG   3                                       ;
    FFFF  000DE                 ;TPA$TARGET
                                U.108:  .WORD   -1                                      ;
    F304  000E0                 ;TPA$TYPE
                                U.112:  .WORD   -3324                                   ;
      01  000E2                 ;TPA$FLAGS2
                                U.113:  .BYTE   1                                       ;
00000001  000E3                 ;TPA$PARAM
                                U.114:  .LONG   1                                       ;
00000000V 000E7                 ;TPA$ACTION
                                U.115:  .LONG   <<VALUE_REQ-U.115>-4>                   ;
00000000* 000EB                 ;TPA$ADDR
                                U.116:  .LONG   <<LIB$GL_TPINDEX-U.116>-4>              ;
00000004  000EF                 ;TPA$MASK
                                U.117:  .LONG   4                                       ;
    FFFF  000F3                 ;TPA$TARGET
                                U.118:  .WORD   -1                                      ;
    F305  000F5                 ;TPA$TYPE
                                U.122:  .WORD   -3323                                   ;
      01  000F7                 ;TPA$FLAGS2
                                U.123:  .BYTE   1                                       ;
00000001  000F8                 ;TPA$PARAM
                                U.124:  .LONG   1                                       ;
```

```
                                00000000V 000FC ;TPA$ACTION
                                                U.125:  .LONG   <<VALUE_REQ-U.125>-4>                   ;
                                00000000* 00100 ;TPA$ADDR
                                                U.126:  .LONG   <<LIB$GL_TPINDEX-U.126>-4>              ;
                                00000005  00104 ;TPA$MASK
                                                U.127:  .LONG   5                                       ;
                                    FFFF  00108 ;TPA$TARGET
                                                U.128:  .WORD   -1                                      ;
                                    F306  0010A ;TPA$TYPE
                                                U.132:  .WORD   -3322                                   ;
                                      01  0010C ;TPA$FLAGS2
                                                U.133:  .BYTE   1                                       ;
                                00000000  0010D ;TPA$PARAM
                                                U.134:  .LONG   0                                       ;
                                00000000V 00111 ;TPA$ACTION
                                                U.135:  .LONG   <<VALUE_REQ-U.135>-4>                   ;
                                00000000* 00115 ;TPA$ADDR
                                                U.136:  .LONG   <<LIB$GL_CRE8FLAGS-U.136>-4>            ;
                                00000040  00119 ;TPA$MASK
                                                U.137:  .LONG   64                                      ;
                                    FFFF  0011D ;TPA$TARGET
                                                U.138:  .WORD   -1                                      ;
                                    F307  0011F ;TPA$TYPE
                                                U.142:  .WORD   -3321                                   ;
                                      01  00121 ;TPA$FLAGS2
                                                U.143:  .BYTE   1                                       ;
                                00000000  00122 ;TPA$PARAM
                                                U.144:  .LONG   0                                       ;
                                00000000V 00126 ;TPA$ACTION
                                                U.145:  .LONG   <<VALUE_REQ-U.145>-4>                   ;
                                00000000* 0012A ;TPA$ADDR
                                                U.146:  .LONG   <<LIB$GL_CRE8FLAGS-U.146>-4>            ;
                                00000080  0012E ;TPA$MASK
                                                U.147:  .LONG   128                                     ;
                                    FFFF  00132 ;TPA$TARGET
                                                U.148:  .WORD   -1                                      ;
                                    97F6  00134 ;TPA$TYPE
                                                U.149:  .WORD   -26634                                  ;
                                      01  00136 ;TPA$FLAGS2
                                                U.150:  .BYTE   1                                       ;
                                0086110C  00137 ;TPA$PARAM
                                                U.151:  .LONG   8786188                                 ;
                                00000000V 0013B ;TPA$ACTION
                                                U.152:  .LONG   <<SYNTAXERR-U.152>-4>                   ;
                                    FFFF  0013F ;TPA$TARGET
                                                U.153:  .WORD   -1                                      ;
                                          00141         .BLKB   3
                                          00144 DATA_STATES::
                                                        .BLKB   0
                                    F300  00144 ;TPA$TYPE
                                                U.159:  .WORD   -3328                                   ;
                                      01  00146 ;TPA$FLAGS2
                                                U.160:  .BYTE   1                                       ;
                                00000000  00147 ;TPA$PARAM
                                                U.161:  .LONG   0                                       ;
                                00000000V 0014B ;TPA$ACTION
                                                U.162:  .LONG   <<VALUE_REQ-U.162>-4>                   ;
```

```
00000000* 0014F ;TPA$ADDR
                U.163:  .LONG   <<LIB$GL_CRE8FLAGS-U.163>-4>                    ;
00000080  00153 ;TPA$MASK
                U.164:  .LONG   128                                             ;
    FFFF  00157 ;TPA$TARGET
                U.165:  .WORD   -1                                              ;
    9301  00159 ;TPA$TYPE
                U.169:  .WORD   -27903                                          ;
      01  0015B ;TPA$FLAGS2
                U.170:  .BYTE   1                                               ;
00000000  0015C ;TPA$PARAM
                U.171:  .LONG   0                                               ;
00000000V 00160 ;TPA$ACTION
                U.172:  .LONG   <<VALUE_REQ-U.172>-4>                           ;
    FFFF  00164 ;TPA$TARGET
                U.173:  .WORD   -1                                              ;
    97F6  00166 ;TPA$TYPE
                U.174:  .WORD   -26634                                          ;
      01  00168 ;TPA$FLAGS2
                U.175:  .BYTE   1                                               ;
0086110C  00169 ;TPA$PARAM
                U.176:  .LONG   8786188                                         ;
00000000V 0016D ;TPA$ACTION
                U.177:  .LONG   <<SYNTAXERR-U.177>-4>                           ;
    FFFF  00171 ;TPA$TARGET
                U.178:  .WORD   -1                                              ;

                        .PSECT  _LIB$KEY0$,NOWRT,  SHR,  PIC,1

      00000 CREATE_KEYS::
                        .BLKB   0
      00000 ;TPA$KEY0
                U.1:    .BLKB   0
 0000* 00000 ;TPA$KEY
                U.3:    .WORD   <U.2-U.1>                                       ;
 0000* 00002 ;TPA$KEY
                U.13:   .WORD   <U.12-U.1>                                      ;
 0000* 00004 ;TPA$KEY
                U.23:   .WORD   <U.22-U.1>                                      ;
 0000* 00006 ;TPA$KEY
                U.33:   .WORD   <U.32-U.1>                                      ;
 0000* 00008 ;TPA$KEY
                U.43:   .WORD   <U.42-U.1>                                      ;
 0000* 0000A ;TPA$KEY
                U.53:   .WORD   <U.52-U.1>                                      ;
      0000C COMPRESS_KEYS::
                        .BLKB   0
      0000C ;TPA$KEY0
                U.68:   .BLKB   0
 0000* 0000C ;TPA$KEY
                U.70:   .WORD   <U.69-U.68>                                     ;
 0000* 0000E ;TPA$KEY
                U.80:   .WORD   <U.79-U.68>                                     ;
 0000* 00010 ;TPA$KEY
                U.90:   .WORD   <U.89-U.68>                                     ;
 0000* 00012 ;TPA$KEY
                U.100:  .WORD   <U.99-U.68>                                     ;
```

```
                                0000* 00014 ;TPA$KEY
                                            U.110:  .WORD   <U.109-U.68>                     ;
                                0000* 00016 ;TPA$KEY
                                            U.120:  .WORD   <U.119-U.68>                     ;
                                0000* 00018 ;TPA$KEY
                                            U.130:  .WORD   <U.129-U.68>                     ;
                                0000* 0001A ;TPA$KEY
                                            U.140:  .WORD   <U.139-U.68>                     ;
                                      0001C DATA_KEYS::
                                            .BLKB   0
                                      0001C ;TPA$KEY0
                                            U.155:  .BLKB   0
                                0000* 0001C ;TPA$KEY
                                            U.157:  .WORD   <U.156-U.155>                    ;
                                0000* 0001E ;TPA$KEY
                                            U.167:  .WORD   <U.166-U.155>                    ;

                                            .PSECT  $PLIT$,NOWRT,NOEXE,2

                  53 45 4C 55 44 4F 4D  00000 P.AAB:  .ASCII  \MODULES\                      ;
                                        00007         .BLKB   1
                           00000007     00008 P.AAA:  .LONG   7                             ;
                           00000000'    0000C         .ADDRESS P.AAB                       ;
                     45 52 4F 46 45 42  00010 P.AAD:  .ASCII  \BEFORE\                      ;
                                        00016         .BLKB   2
                           00000006     00018 P.AAC:  .LONG   6                             ;
                           00000000'    0001C         .ADDRESS P.AAD                       ;
               53 53 45 52 50 4D 4F 43  00020 P.AAF:  .ASCII  \COMPRESS\                    ;
                           00000008     00028 P.AAE:  .LONG   8                             ;
                           00000000'    0002C         .ADDRESS P.AAF                       ;
                     45 54 41 45 52 43  00030 P.AAH:  .ASCII  \CREATE\                      ;
                                        00036         .BLKB   2
                                 00000006  00038 P.AAG:  .LONG                              ;
                                 00000000' 0003C         .ADDRESS P.AAH                     ;
45 43 4E 45 52 45 46 45 52 5F 53 53 4F 52 43  00040 P.AAJ:  .ASCII  \CROSS_REFERENCE\       ;
                                        0004F         .BLKB   1
                                 0000000F  00050 P.AAI:  .LONG   15                         ;
                                 00000000' 00054         .ADDRESS P.AAJ                     ;
                              41 54 41 44  00058 P.AAL:  .ASCII  \DATA\                     ;
                                 00000004  0005C P.AAK:  .LONG   4                          ;
                                 00000000' 00060         .ADDRESS P.AAL                     ;
                           45 54 45 4C 45 44  00064 P.AAN:  .ASCII  \DELETE\                ;
                                        0006A         .BLKB   2
                                 00000006  0006C P.AAM:  .LONG   6                          ;
                                 00000000' 00070         .ADDRESS P.AAN                     ;
                        54 43 41 52 54 58 45  00074 P.AAP:  .ASCII  \EXTRACT\               ;
                                        0007B         .BLKB   1
                                 00000007  0007C P.AAO:  .LONG   7                          ;
                                 00000000' 00080         .ADDRESS P.AAP                     ;
                              4C 4C 55 46  00084 P.AAR:  .ASCII  \FULL\                     ;
                                 00000004  00088 P.AAQ:  .LONG   4                          ;
                                 00000000' 0008C         .ADDRESS P.AAR                     ;
                        53 4C 41 42 4F 4C 47  00090 P.AAT:  .ASCII  \GLOBALS\               ;
                                        00097         .BLKB   1
                                 00000007  00098 P.AAS:  .LONG   7                          ;
                                 00000000' 0009C         .ADDRESS P.AAT                     ;
                              50 4C 45 48  000A0 P.AAV:  .ASCII  \HELP\                     ;
```

```
                                00000004  000A4 P.AAU:   .LONG   4
                                00000000' 000A8        .ADDRESS P.AAV
                    59  52  4F  54  53  49  48  000AC P.AAX:   .ASCII  \HISTORY\
                                            000B3      .BLKB   1
                                00000007  000B4 P.AAW:   .LONG   7
                                00000000' 000B8        .ADDRESS P.AAX
                        54  52  45  53  4E  49  000BC P.AAZ:   .ASCII  \INSERT\
                                            000C2      .BLKB   2
                                00000006  000C4 P.AAY:   .LONG   6
                                00000000' 000C8        .ADDRESS P.AAZ
                            54  53  49  4C  000CC P.ABB:   .ASCII  \LIST\
                                00000004  000D0 P.ABA:   .LONG   4
                                00000000' 000D4        .ADDRESS P.ABB
                            47  4F  4C      000D8 P.ABD:   .ASCII  \LOG\
                                            000DB      .BLKB   1
                                00000003  000DC P.ABC:   .LONG   3
                                00000000' 000E0        .ADDRESS P.ABD
                        4F  52  43  41  4D  000E4 P.ABF:   .ASCII  \MACRO\
                                            000E9      .BLKB   3
                                00000005  000EC P.ABE:   .LONG   5
                                00000000' 000F0        .ADDRESS P.ABF
                        53  45  4D  41  4E  000F4 P.ABH:   .ASCII  \NAMES\
                                            000F9      .BLKB   3
                                00000005  000FC P.ABG:   .LONG   5
                                00000000' 00100        .ADDRESS P.ABH
                    54  43  45  4A  42  4F  00104 P.ABJ:   .ASCII  \OBJECT\
                                            0010A      .BLKB   2
                                00000006  0010C P.ABI:   .LONG   6
                                00000000' 00110        .ADDRESS P.ABJ
                            59  4C  4E  4F  00114 P.ABL:   .ASCII  \ONLY\
                                00000004  00118 P.ABK:   .LONG   4
                                00000000' 0011C        .ADDRESS P.ABL
                    54  55  50  54  55  4F  00120 P.ABN:   .ASCII  \OUTPUT\
                                            00126      .BLKB   2
                                00000006  00128 P.ABM:   .LONG   6
                                00000000' 0012C        .ADDRESS P.ABN
                        45  56  4F  4D  45  52  00130 P.ABP:   .ASCII  \REMOVE\
                                            00136      .BLKB   2
                                00000006  00138 P.ABO:   .LONG   6
                                00000000' 0013C        .ADDRESS P.ABP
                    45  43  41  4C  50  45  52  00140 P.ABR:   .ASCII  \REPLACE\
                                            00147      .BLKB   1
                                00000007  00148 P.ABQ:   .LONG   7
                                00000000' 0014C        .ADDRESS P.ABR
43  52  41  45  53  5F  45  56  49  54  43  45  4C  45  53  00150 P.ABT:   .ASCII  \SELECTIVE_SEARCH\
                                                    48  0015F
                                00000010  00160 P.ABS:   .LONG   16
                                00000000' 00164        .ADDRESS P.ABT
                            45  52  41  48  53  00168 P.ABV:   .ASCII  \SHARE\
                                            0016D      .BLKB   3
                                00000005  00170 P.ABU:   .LONG   5
                                00000000' 00174        .ADDRESS P.ABV
                            45  43  4E  49  53  00178 P.ABX:   .ASCII  \SINCE\
                                            0017D      .BLKB   3
                                00000005  00180 P.ABW:   .LONG   5
                                00000000' 00184        .ADDRESS P.ABX
                    45  5A  45  45  55  51  53  00188 P.ABZ:   .ASCII  \SQUEEZE\
```

```
                                                      0018F             .BLKB    1
                                           00000007   00190 P.ABY:     .LONG    7
                                           00000000'  00194            .ADDRESS P.ABZ
                                  54  58   45   54    00198 P.ACB:     .ASCII   \TEXT\
                                           00000004   0019C P.ACA:     .LONG    4
                                           00000000'  001A0            .ADDRESS P.ACB
                              48  54  44   49   57    001A4 P.ACD:     .ASCII   \WIDTH\
                                                      001A9            .BLKB    3
                                           00000005   001AC P.ACC:     .LONG    5
                                           00000000'  001B0            .ADDRESS P.ACD
                                                31 50 001B4 P.ACF:     .ASCII   \P1\
                                                      001B6            .BLKB    2
                                           00000002   001B8 P.ACE:     .LONG    2
                                           00000000'  001BC            .ADDRESS P.ACF
                                                32 50 001C0 P.ACH:     .ASCII   \P2\
                                                      001C2            .BLKB    2
                                           00000002   001C4 P.ACG:     .LONG    2
                                           00000000'  001C8            .ADDRESS P.ACH
                    00  4C  4F  42  4D  59  53  06    001CC P.ACI:     .ASCII   <6>\SYMBOL\<0>
                    00  00  45  55  4C  41  56  05    001D4 P.ACJ:     .ASCII   <5>\VALUE\<0><0>
                    00  45  4C  55  44  4F  4D  06    001DC P.ACK:     .ASCII   <6>\MODULE\<0>
                                    4C  4C  41  03    001E4 P.ACL:     .ASCII   <3>\ALL\
                    00  00  00  45  4E  4F  4E  04    001E8 P.ACM:     .ASCII   <4>\NONE\<0><0><0>
    00  3A  54  55  50  54  55  4F  24  53  59  53    001F0 P.ACN:     .ASCII   \SYS$OUTPUT:\<0>
                    00  00  4C  4F  42  4D  59  53    001FC P.ACO:     .ASCII   \SYMBOL\<0><0>
                    00  00  00  45  55  4C  41  56    00204 P.ACP:     .ASCII   \VALUE\<0><0><0>
                    00  00  45  4C  55  44  4F  4D    0020C P.ACQ:     .ASCII   \MODULE\<0><0>
                                    00  4C  4C  41    00214 P.ACR:     .ASCII   \ALL\<0>
                                    45  4E  4F  4E    00218 P.ACS:     .ASCII   \NONE\

                                                                       .PSECT   $OWN$,NOEXE,2

00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00000 QUAL_TABLE:
                                                                       .ADDRESS SD_BEFORE, SD_COMPRESS, SD_CREATE, -
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00018               SD_CROSS_REFERENCE, SD_DATA, SD_DELETE, -
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00030               SD_EXTRACT, SD_FULL, SD_GLOBALS, SD_HELP, -
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00048               SD_HISTORY, SD_LIST, SD_LOG, SD_MACRO, -
                                                                                SD_NAMES, SD_OBJECT, SD_ONLY, SD_REMOVE, -
                                                                                SD_SHARE, SD_SINCE, SD_TEXT, SD_REPLACE, -
                                                                                SD_INSERT, SD_SQUEEZE
0F  02  16  0B  1D  03  11  10  09  08  23  1C  07  06  20  00060 BIT_TABLE:
                                                                       .BYTE    32, 6, 7, 28, 35, 8, 9, 16, 17, 3, 29, -
                    13  0A  0D  04  21  05  0C  1A  01  0006F                   11, 22, 2, 15, 1, 26, 12, 5, 33, 4, 13, -
                                                                                10, 19
0000000V 0000000V 0000000V 0000000V 0000000V 0000000V 00078 CALL_TABLE:
                                                                       .ADDRESS BEFORE_DATE, COMPRESSFILE, CREATELIB, -
                                                  0000000V 00090                SELECTCROSREF, DATAREDUCE, DELETEMODULES, -
                                                                                EXTRACTFILE
                                       00000000   00000000  00094      .LONG    0, 0
                                                  0000000V 0009C       .ADDRESS SET_HELP_TYPE
                                                  00000000  000A0      .LONG    0
                                                  0000000V 000A4       .ADDRESS SETLISTFILE
                                                  00000000  000A8      .LONG    0
                                                  0000000V 000AC       .ADDRESS SET_MACRO_TYPE
                                                  00000000  000B0      .LONG    0
0000000V 0000000V 0000000V 0000000V 0000000V 0000000V 000B4            .ADDRESS SET_OBJECT_TYPE, LISTONLYMODS, -
                                                                                REMOVESYMBOLS, SET_SHR_TYPE, SINCE_DATE, -
```

```
                                                        SET_TEXT_TYPE
                00000000  00000000  00000000  000CC              .LONG   0, 0, 0
                                    00000006  000D8 CRF_TABLE:
                                                       .LONG   6
                                    00000000' 000DC            .ADDRESS P.ACI
                                    00000000  000E0            .LONG   0
                                    00000000' 000E4            .ADDRESS P.ACJ
                                    00000001  000E8            .LONG   1
                                    00000000' 000EC            .ADDRESS P.ACK
                                    00000002  000F0            .LONG   2
                                    00000004  000F4 OPT_TABLE:
                                                       .LONG   4
                                    00000000' 000F8            .ADDRESS P.ACL
                                    00000000  000FC            .LONG   0
                                    00000000' 00100            .ADDRESS P.ACM
                                    00000001  00104            .LONG   1
                                        0000  00108 LIST_DESC:
                                                       .WORD   0
                                          00  0010A            .BYTE   0
                                          02  0010B            .BYTE   2
                                    00000000  0010C            .LONG   0
                                    00000001  00110 SQUEEZE_FLAG:
                                                       .LONG   1
                                              00114 DEF_LIB_EXTN:
                                                       .BLKB   8
                                              0011C DEF_FIL_EXTN:
                                                       .BLKB   8
                                              00124 LASTFDB:.BLKB   4
                                              00128 PREVFDB:.BLKB   4
                                    0000000B  0012C SYSOUTPUTDESC:
                                                       .LONG   11
                                    00000000' 00130            .ADDRESS P.ACN
                                        0000  00134 MODULENAMEDESC:
                                                       .WORD   0
                                          00  00136            .BYTE   0
                                          02  00137            .BYTE   2
                                    00000000  00138            .LONG   0
                                              0013C CLIWORKPTR:
                                                       .BLKB   4
                                              00140 FILESNOTFOUND:
                                                       .BLKB   4
                                        0000  00144 TOKEN_DESC:
                                                       .WORD   0
                                          00  00146            .BYTE   0
                                          02  00147            .BYTE   2
                                    00000000  00148            .LONG   0
                                    00000006  0014C CRF_BYSYMBOL:
                                                       .LONG   6
                                    00000000' 00150            .ADDRESS P.ACO
                                    00000005  00154 CRF_BYVALUE:
                                                       .LONG   5
                                    00000000' 00158            .ADDRESS P.ACP
                                    00000006  0015C CRF_BYMODULE:
                                                       .LONG   6
                                    00000000' 00160            .ADDRESS P.ACQ
                                    00000000  00164 CRF_BYNONE:
                                                       .LONG   0
```

```
00000003  00168 ALL_OPTIONS:
                        .LONG   3
00000000' 0016C         .ADDRESS P.ACR
00000004  00170 NO_OPTIONS:
                        .LONG   4
00000000' 00174         .ADDRESS P.ACS
00000000  00178 END_OPTIONS:
                        .LONG   0
          0017C TPA_BLOCK:
                        .BLKB   36
          001A0 TPA_DESC:
                        .BLKB   4

                        .PSECT  $GLOBAL$,NOEXE,2

          00000 LIB$GL_TPINDEX::
                        .BLKB   4
          00004 LIB$GL_VALREQ::
                        .BLKB   4
          00008 LIB$GL_CREFLAGS::
                        .BLKB   4

                SD_MODULES=             P.AAA
                SD_BEFORE=              P.AAC
                SD_COMPRESS=            P.AAE
                SD_CREATE=              P.AAG
                SD_CROSS_REFERENCE=     P.AAI
                SD_DATA=                P.AAK
                SD_DELETE=              P.AAM
                SD_EXTRACT=             P.AAO
                SD_FULL=                P.AAQ
                SD_GLOBALS=             P.AAS
                SD_HELP=                P.AAU
                SD_HISTORY=             P.AAW
                SD_INSERT=              P.AAY
                SD_LIST=                P.ABA
                SD_LOG=                 P.ABC
                SD_MACRO=               P.ABE
                SD_NAMES=               P.ABG
                SD_OBJECT=              P.ABI
                SD_ONLY=                P.ABK
                SD_OUTPUT=              P.ABM
                SD_REMOVE=              P.ABO
                SD_REPLACE=             P.ABQ
                SD_SELECTIVE_SEARCH=    P.ABS
                SD_SHARE=               P.ABU
                SD_SINCE=               P.ABW
                SD_SQUEEZE=             P.ABY
                SD_TEXT=                P.ACA
                SD_WIDTH=               P.ACC
                SD_P1=                  P.ACE
                SD_P2=                  P.ACG
                        .EXTRN  LIB$CVT_DTB, CLI$GET_VALUE
                        .EXTRN  CLI$PRESENT, GETFILNAMDESC
                        .EXTRN  LIB_GET_MEM, LIB$CVT_TIME
                        .EXTRN  LIB$FILE_SCAN, LIB$TPARSE
                        .EXTRN  LIB$LOOKUP_KEY, LIB_FREE_MEM
```

```
                                        .EXTRN  LIB$BEFORE_DATE
                                        .EXTRN  LIB$SINCE_DATE, LIB$AL_CREOPTS
                                        .EXTRN  LIB$GL_CREBFLAGS
                                        .EXTRN  LIB$GL_LISTWID, LIB$GL_TYPE
                                        .EXTRN  LIB$GL_MODLISL, LIB$GL_MODXTRL
                                        .EXTRN  LIB$GL_MODUPDL, LIB$GL_OBJSYRML
                                        .EXTRN  LIB$GL_DELMODL, LIB$GL_CTLMSK
                                        .EXTRN  LIB$GL_LIBFDB, LIB$GL_OUTFDB
                                        .EXTRN  LIB$GL_LISFDB, LIB$GL_TMPFDB
                                        .EXTRN  LIB$GL_INPLIST, LIB_MAC_DEFEXT
                                        .EXTRN  LIB_OBJ_DEFEXT, LIB_HLP_DEFEXT
                                        .EXTRN  LIB_TXT_DEFEXT, LIB_SHR_DEFEXT

                                        .PSECT  $CODE$,NOWRT,2

                                0FFC 00000      .ENTRY  LIB_GET_COMMAND, Save R2,R3,R4,R5,R6,R7,R8,-; 0753
                                                        R9,R10,R11
                          5B    0000G   CF 9E 00002     MOVAB   LIB$GL_OBJSYRML, R11
                          5A    0000G   CF 9E 00007     MOVAB   LIB$GL_MODUPDL, R10
                          59    0000G   CF 9E 0000C     MOVAB   LIB$GL_MODXTRL, R9
                          58    0000G   CF 9E 00011     MOVAB   LIB$GL_MODLISL, R8
                          57    0000'   CF 9E 00016     MOVAB   TOKEN_DESC, R7
                          56    0000G   CF 9E 0001B     MOVAB   LIB$GL_CTLMSK, R6
                    E0    A7    0000G   CF 9E 00020     MOVAB   LIB$GL_INPLIST, LASTFDB                 0774
                    E4    A7    0000G   CF 9E 00026     MOVAB   LIB$GL_INPLIST, PREVFDB                 0775
                                0000G   CF D4 0002C     CLRL    LIB$GL_INPLIST                          0776
                                0000G   CF 9F 00030     PUSHAB  LIB_OBJ_DEFEXT                          0782
                                        7E D4 00034     CLRL    -(SP)
                    0000V       CF          02 FB 00036 CALLS   #2, SET_LIB_TYPE
                                68          68 9E 0003B MOVAB   LIB$GL_MODLISL, LIB$GL_MODLISL          0787
                          04    A8          68 9E 0003E MOVAB   LIB$GL_MODLISL, LIB$GL_MODLISL+4        0788
                                69          69 9E 00042 MOVAB   LIB$GL_MODXTRL, LIB$GL_MODXTRL          0789
                          04    A9          69 9E 00045 MOVAB   LIB$GL_MODXTRL, LIB$GL_MODXTRL+4        0790
                                6A          6A 9E 00049 MOVAB   LIB$GL_MODUPDL, LIB$GL_MODUPDL          0791
                          04    AA          6A 9E 0004C MOVAB   LIB$GL_MODUPDL, LIB$GL_MODUPDL+4        0792
                                6B          6B 9E 00050 MOVAB   LIB$GL_OBJSYRML, LIB$GL_OBJSYRML        0793
                          04    AB          6B 9E 00053 MOVAB   LIB$GL_OBJSYRML, LIB$GL_OBJSYRML+4      0794
                    0000G       CF  0000G   CF 9E 00057 MOVAB   LIB$GL_DELMODL, LIB$GL_DELMODL          0795
                    0000G       CF  0000G   CF 9E 0005E MOVAB   LIB$GL_DELMODL, LIB$GL_DELMODL+4        0796
                                        52 D4 00065     CLRL    I                                       0801
                              FEBC C742     DD 00067 1$:    PUSHL   QUAL_TABLE[I]                       0802
                    0000G       CF          01 FB 0006C     CALLS   #1, CLI$PRESENT
                                53  FF1C C742 9A 00071     MOVZBL  BIT_TABLE[I], R3
                                51          50 D0 00077     MOVL    R0, R1
                66        01    53          51 F0 0007A     INSV    R1, R3, #1, LIB$GL_CTLMSK
                                0B          51 E9 0007F     BLBC    R1, 2$
                                50  FF34 C742 D0 00082     MOVL    CALL_TABLE[I], R0                   0807
                                            03 13 00088     BEQL    2$
                                60          00 FB 0008A     CALLS   #0, (R0)                            0809
                          D6    52          17 F3 0008D 2$: AOBLEQ  #23, I, 1$                          0802
                          07  02    A6      03 E1 00091     BBC     #3, LIB$GL_CTLMSK+2, 3$             0811
                    0000V       CF          00 FB 00096     CALLS   #0, SETSQUEEZE                      0813
                                            05 11 0009B     BRB     4$
                    0000V       CF          00 FB 0009D 3$: CALLS   #0, CLEARSQUEEZE                    0815
                                0000'   CF  9F 000A2 4$:    PUSHAB  SD_WIDTH                            0817
                    0000G       CF          01 FB 000A6     CALLS   #1, CLI$PRESENT
                                05          50 E9 000AB     BLBC    R0, 5$
```

```
               0000V  CF            00 FB 000AE        CALLS   #0, SETLISTWIDTH                    ; 0819
                                    57 DD 000B3 5$:    PUSHL   R7                                  ; 0824
                            0000'   CF 9F 000B5        PUSHAB  SD_P1                               ;
               0000G  CF            02 FB 000B9        CALLS   #2, CLI$GET_VALUE                   ;
               0000V  CF            00 FB 000BE        CALLS   #0, INPUT1FILE                      ; 0825
                                    57 DD 000C3        PUSHL   R7                                  ; 0830
                            0000'   CF 9F 000C5        PUSHAB  SD_OUTPUT                           ;
               0000G  CF            02 FB 000C9        CALLS   #2, CLI$GET_VALUE                   ;
               0000V  CF            00 FB 000CE        CALLS   #0, OUTPUTFILE                      ; 0831
                              FC    A7 D4 000D3        CLRL    FILESNOTFOUND                       ; 0833
        48            66            06 E0 000D6        BBS     #6, LIB$GL_CTLMSK, 8$               ; 0835
        43       04   A6            03 E0 000DA        BBS     #3, LIB$GL_CTLMSK+4, 8$             ; 0836
        3E       01   A6            01 E0 000DF        BBS     #1, LIB$GL_CTLMSK+1, 8$             ; 0837
        39       03   A6            04 E0 000E4        BBS     #4, LIB$GL_CTLMSK+3, 8$             ; 0838
                                    57 DD 000E9 6$:    PUSHL   R7                                  ; 0840
                            0000'   CF 9F 000EB        PUSHAB  SD_P2                               ;
               0000G  CF            02 FB 000EF        CALLS   #2, CLI$GET_VALUE                   ;
                      2B            50 E9 000F4        BLBC    R0, 8$                              ;
                 02   A6            04 8A 000F7        BICB2   #4, LIB$GL_CTLMSK+2                 ; 0842
                            0000'   CF 9F 000FB        PUSHAB  SD_MODULES                          ; 0843
               0000G  CF            01 FB 000FF        CALLS   #1, CLI$PRESENT                     ;
                      05            50 E9 00104        BLBC    R0, 7$                              ;
               0000v  CF            00 FB 00107        CALLS   #0, SETMODULENAME                   ; 0845
                                    57 DD 0010C 7$:    PUSHL   R7                                  ; 0846
               0000V  CF            01 FB 0010E        CALLS   #1, INPUT2FILE                      ;
        D1       02   A6            02 E1 00113        BBC     #2, LIB$GL_CTLMSK+2, 6$             ; 0847
                      50      E0    A7 D0 00118        MOVL    LASTFDB, R0                         ; 0848
                 04   A0            01 88 0011C        BISB2   #1, 4(R0)                           ;
                                    C7 11 00120        BRB     6$                                  ; 0840
                      50    0000G   CF D0 00122 8$:    MOVL    LIB$GL_INPLIST, R0                  ; 0851
                                    2A 12 00127        BNEQ    9$                                  ;
                              FC    A7 D5 00129        TSTL    FILESNOTFOUND                       ; 0852
                                    25 13 0012C        BEQL    9$                                  ;
                                    66 95 0012E        TSTB    LIB$GL_CTLMSK                       ; 0853
                                    5E 19 00130        BLSS    14$                                 ;
                      1D      01    A6 E8 00132        BLBS    LIB$GL_CTLMSK+1, 9$                 ; 0855
        18       01   A6            01 E0 00136        BBS     #1, LIB$GL_CTLMSK+1, 9$             ; 0856
        13       01   A6            03 E0 0013B        BBS     #3, LIB$GL_CTLMSK+1, 9$             ; 0857
        0E       01   A6            04 E0 00140        BBS     #4, LIB$GL_CTLMSK+1, 9$             ; 0858
        09       04   A6            03 E0 00145        BBS     #3, LIB$GL_CTLMSK+4, 9$             ; 0859
        05            66            06 E0 0014A        BBS     #6, LIB$GL_CTLMSK, 9$               ; 0860
        3D       03   A6            04 E1 0014E        BBC     #4, LIB$GL_CTLMSK+3, 14$            ; 0861
                                    50 D5 00153 9$:    TSTL    R0                                  ; 0864
                                    09 13 00155        BEQL    10$                                 ;
        04       01   A6            02 E0 00157        BBS     #2, LIB$GL_CTLMSK+1, 10$            ; 0865
                 01   A6            24 88 0015C        BISB2   #36, LIB$GL_CTLMSK+1                ; 0866
                      50    0000G   CF D0 00160 10$:   MOVL    LIB$GL_LIBFDB, R0                   ; 0870
   08   A0       D0   A7            08 28 00165        MOVC3   #8, DEF_LIB_EXTN, 8(R0)             ;
        09       01   A6            02 E0 0016B        BBS     #2, LIB$GL_CTLMSK+1, 11$            ; 0871
                      05      01    A6 E8 00170        BLBS    LIB$GL_CTLMSK+1, 11$                ;
        09       01   A6            04 E1 00174        BBC     #4, LIB$GL_CTLMSK+1, 12$            ; 0872
                            0000G   CF 9F 00179 11$:   PUSHAB  LIB$GL_TMPFDB                       ; 0873
               0000V  CF            01 FB 0017D        CALLS   #1, ALLOCATE_FDB                    ;
                            0000'   CF D5 00182 12$:   TSTL    LIB$GL_CREFLAGS                     ; 0875
                                    04 12 00186        BNEQ    13$                                 ;
                 03   A6            10 8A 00188        BICB2   #16, LIB$GL_CTLMSK+3                ; 0876
                      50            01 D0 0018C 13$:   MOVL    #1, R0                              ; 0877
```

```
                    04 0018F                RET
                50  D4 00190 14$:           CLRL    R0                              ; 0878
                    04 00192                RET
                                                                                    ;

; Routine Size:  403 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  413          0879  1 %SBTTL 'inputlfile';
;  414          0880  1
;  415          0881  1 ROUTINE inputlfile  =
;  416          0882  2 BEGIN
;  417          0883  2 !++
;  418          0884  2 ! This routine is called by CLI when the library filename
;  419          0885  2 ! is processed.
;  420          0886  2 !
;  421          0887  2 ! Implicit inputs:
;  422          0888  2 !
;  423          0889  2 !       token_desc      Address of the  descriptor
;  424          0890  2 !
;  425          0891  2 ! Outputs:
;  426          0892  2 !
;  427          0893  2 !       An fdb is allocated for the library file and the information is filled
;  428          0894  2 !       in.  If not creating the library, open the file to ensure it exists,
;  429          0895  2 !       and determine the library type expected if an explicit extension was
;  430          0896  2 !       specified.
;  431          0897  2 !
;  432          0898  2 !--
;  433          0899  2 LOCAL
;  434          0900  2     status,
;  435          0901  2     openstatus,
;  436          0902  2     filenamestring,
;  437          0903  2     lfab : BBLOCK [fab$c_bln];
;  438          0904  2
;  439          0905  2 allocate_fdb (lib$gl_libfdb);                          !Allocate a new FDB for library
;  440          0906  2 lib_get_mem(.token_desc[dsc$w_length], filenamestring);
;  441          0907  2 ch$move (.token_desc[dsc$w_length], .token_desc[dsc$a_pointer], .filenamestring);
;  442          0908  2 lib$gl_libfdb[fdb$l_namdesc] = .token_desc[dsc$w_length];
;  443          0909  2 lib$gl_libfdb[fdb$l_namdesc] + 4 = .filenamestring;
;  444          0910  2 CH$MOVE (dsc$c_s_bln, def_lib_extn, lib$gl_libfdb [fdb$l_defext]);
;  445          0911  2 IF NOT .lib$gl_ctlmsk [lib$v_create]                    !Unless creating the library
;  446          0912  2 THEN
;  447          0913  3     BEGIN
;  448          0914  3     BIND
;  449          0915  3         filenamedesc = lib$gl_libfdb [fdb$l_namdesc] : BBLOCK,
;  450          0916  3         namblk = lib$gl_libfdb [fdb$t_nam] : BBLOCK;
;  451          0917  3
;  452        P 0918  3     $FAB_INIT (FAB = lfab,
;  453        P 0919  3                FNS = .token_desc [dsc$w_length],
;  454        P 0920  3                FNA = .token_desc [dsc$a_pointer],
;  455        P 0921  3                NAM = namblk,
;  456        P 0922  3                DNS = .def_lib_extn [dsc$w_length],
;  457          0923  3                DNA = .def_lib_extn [dsc$a_pointer]);
;  458          0924  4     IF NOT (openstatus = $OPEN (FAB = lfab))
;  459          0925  3         AND .openstatus NEQ rms$_flk                   !LBR will wait if locked
;  460          0926  4         THEN BEGIN
;  461          0927  4             getfilnamdesc (lfab, filenamedesc);        !Get string descriptor for filename
;  462          0928  4             SIGNAL_STOP (lib$_openin, 1, filenamedesc, .openstatus, .lfab [fab$l_stv]);
;  463          0929  3             END;
;  464          0930  3     getfilnamdesc (lfab, filenamedesc);
;  465          0931  4     IF NOT (IF NOT .openstatus THEN true
;  466          0932  4                 ELSE (status = $CLOSE (FAB = lfab)))
;  467          0933  3         THEN SIGNAL (lib$_closein, 1, lib$gl_libfdb [fdb$l_namdesc], .status,
;  468          0934  3                      .lfab [fab$l_stv]);
;  469          0935  3     IF .namblk [nam$v_exp_type]                        !If explicit extension given
```

```
;  470     0936  3          AND .namblk [nam$b_rsl] NEQ 0                    ! and resultant string exists
;  471     0937  3          AND .lib$gl_type EQL lbr$c_typ_unk               ! and library type was
;  472     0938  4      THEN BEGIN                                           ! not set with a qualifier
;  473     0939  4      !
;  474     0940  4      ! Lookup the extension in the table. If its a known extension, change
;  475     0941  4      ! the library type to match the extension.
;  476     0942  4      !
;  477     0943  4          LOCAL
;  478     0944  4              ext_start,
;  479     0945  4              ext_end;
;  480     0946  4
;  481     0947  4          IF CH$FAIL(ext_start = CH$FIND_CH (.namblk [nam$b_rsl], !Find end of directory
;  482     0948  4                                  .namblk [nam$l_rsa],
;  483     0949  4                                  %ASCII ']'))
;  484     0950  4              THEN IF CH$FAIL(ext_start = CH$FIND_CH (.namblk [nam$b_rsl], !Find end of directory
;  485     0951  4                                  .namblk [nam$l_rsa],
;  486     0952  4                                  %ASCII '>'))
;  487     0953  4                   THEN ext_start = .namblk[nam$l_rsa];
;  488     0954  4          ext_start = CH$FIND_CH (.namblk [nam$b_rsl] - (.ext_start - .namblk [nam$l_rsa]),
;  489     0955  4                                  .ext_start, %ASCII '.'); !Find start of extension
;  490     0956  4          ext_end = CH$FIND_CH (.namblk [nam$b_rsl] - (.ext_start - .namblk [nam$l_rsa]),
;  491     0957  4                                  .ext_start, %ASCII ';');
;  492     0958  4          INCRU i FROM 0 TO lbr$c_typ_decmx - 1            !Look at all the extensions
;  493     0959  5          DO BEGIN
;  494     0960  5              BIND
;  495     0961  5                  curdesc = lib_obj_defext + .i*(2 * dsc$c_s_bln) : BBLOCK; !Name descriptor
;  496     0962  5
;  497     0963  5              IF CH$EQL (.ext_end - .ext_start, .ext_start,
;  498     0964  5                         .curdesc [dsc$w_length], .curdesc [dsc$a_pointer])
;  499     0965  6                  THEN BEGIN
;  500     0966  6                       set_lib_type (.i + 1, curdesc);     !Set the library type
;  501     0967  6                       EXITLOOP;                           !and we are done
;  502     0968  5                       END;
;  503     0969  4                  END;
;  504     0970  3              END;
;  505     0971  2      END;
;  506     0972  2
;  507     0973  2      ! If the library type is still undefined at this point, default it to "object":
;  508     0974  2      !
;  509     0975  2      IF .lib$gl_type EQL lbr$c_typ_unk THEN lib$gl_type = lib$s_object;
;  510     0976  2      RETURN true
;  511     0977  1      END;                                                 !Of INPUT1FILE


                                                    .EXTRN  SYS$OPEN, SYS$CLOSE

                                    0FFC 00000 INPUT1FILE:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 0881
                              5B    0000G  CF  9E 00002     MOVAB   LIB$GL_LIBFDB, R11                  ;
                              5A    0000G  CF  9E 00007     MOVAB   LIB$GL_TYPE, R10                    ;
                              59    0000'  CF  9E 0000C     MOVAB   TOKEN_DESC, R9                      ;
                              5E      AC   AE  9E 00011     MOVAB   -84(SP), SP                         ;
                                           5B  DD 00015     PUSHL   R11                                 ; 0905
                           C000V  CF        01  FB 00017     CALLS   #1, ALLOCATE_FDB                    ;
                                           5E  DD 0001C     PUSHL   SP                                  ; 0906
                                  7E       69  3C 0001E     MOVZWL  TOKEN_DESC, -(SP)                   ;
```

```
                      0000G  CF             02  FB 00021         CALLS   #2, LIB_GET_MEM                          :
                             58             69  3C 00026         MOVZWL  TOKEN_DESC, R8                           : 0907
             00  BE      04  B9             58  28 00029         MOVC3   R8, @TOKEN_DESC+4, @FILENAMESTRING       :
                             56             6B  D0 0002F         MOVL    LIB$GL_LIBFDB, R6                        : 0908
                         10  A6             58  D0 00032         MOVL    R8, 16(R6)                               :
                         14  A6             6E  D0 00036         MOVL    FILENAMESTRING, 20(R6)                   : 0909
             08  A6      D0  A9             08  28 0003A         MOVC3   #8, DEF_LIB_EXTN, 8(R6)                  : 0910
                                    0000G   CF  95 00040         TSTB    LIB$GL_CTLMSK                            : 0911
                                            03  18 00044         BGEQ    1$                                       :
                                          012E  31 00046         BRW     12$                                      :
                             57         10  A6  9E 00049 1$:     MOVAB   16(R6), R7                               : 0915
                             56         40  A6  9E 0004D         MOVAB   64(R6), R6                               : 0916
   0050  8F              00  6E             00  2C 00051         MOVC5   #0, (SP), #0, #80, $RMS_PTR              : 0923
                                        04  AE     00058                                                          :
                         04  AE       5003  8F  B0 0005A         MOVW    #20483, $RMS_PTR                         :
                         1A  AE             02  90 00060         MOVB    #2, $RMS_PTR+22                          :
                         23  AE             02  90 00064         MOVB    #2, $RMS_PTR+31                          :
                         2C  AE             56  D0 00068         MOVL    R6, $RMS_PTR+40                          :
                         30  AE         04  A9  D0 0006C         MOVL    TOKEN_DESC+4, $RMS_PTR+44                :
                         34  AE         D4  A9  D0 00071         MOVL    DEF_LIB_EXTN+4, $RMS_PTR+48              :
                         38  AE             58  90 00076         MOVB    R8, $RMS_PTR+52                          :
                         39  AE         D0  A9  90 0007A         MOVB    DEF_LIB_EXTN, $RMS_PTR+53                :
                                        04  AE  9F 0007F         PUSHAB  LFAB                                     : 0924
                  00000000G  00             01  FB 00082         CALLS   #1, SYS$OPEN                             :
                             52             50  D0 00089         MOVL    R0, OPENSTATUS                           :
                             29             52  E8 0008C         BLBS    OPENSTATUS, 2$                           :
                  0001828A   8F             52  D1 0008F         CMPL    OPENSTATUS, #98954                       : 0925
                             20             13     00096         BEQL    2$                                       :
                                            57  DD 00098         PUSHL   R7                                       : 0927
                                        08  AE  9F 0009A         PUSHAB  LFAB                                     :
                      0000G  CF             02  FB 0009D         CALLS   #2, GETFILNAMDESC                        :
                                        10  AE  DD 000A2         PUSHL   LFAB+12                                  : 0928
                                            52  DD 000A5         PUSHL   OPENSTATUS                               :
                                            57  DD 000A7         PUSHL   R7                                       :
                                            01  DD 000A9         PUSHL   #1                                       :
                                  00861098  8F  DD 000AB         PUSHL   #8786072                                 :
                  00000000G  00             05  FB 000B1         CALLS   #5, LIB$STOP                             :
                                            57  DD 000B8 2$:     PUSHL   R7                                       : 0930
                                        08  AE  9F 000BA         PUSHAB  LFAB                                     :
                      0000G  CF             02  FB 000BD         CALLS   #2, GETFILNAMDESC                        :
                             25             52  E9 000C2         BLBC    OPENSTATUS, 3$                           : 0931
                                        04  AE  9F 000C5         PUSHAB  LFAB                                     : 0932
                  00000000G  00             01  FB 000C8         CALLS   #1, SYS$CLOSE                            :
                             18             50  E8 000CF         BLBS    STATUS, 3$                               :
                                        10  AE  DD 000D2         PUSHL   LFAB+12                                  : 0934
                                            50  DD 000D5         PUSHL   STATUS                                   : 0933
                 7E          6B             10  C1 000D7         ADDL3   #16, LIB$GL_LIBFDB, -(SP)                :
                                            01  DD 000DB         PUSHL   #1                                       :
                                  00861050  8F  DD 000DD         PUSHL   #8786000                                 :
                  00000000G  00             05  FB 000E3         CALLS   #5, LIB$SIGNAL                           :
                 7F      34  A6             01  E1 000EA 3$:     BBC     #1, 52(R6), 10$                          : 0935
                                        03  A6  95 000EF         TSTB    3(R6)                                    : 0936
                                            7A  13 000F2         BEQL    10$                                      :
                                            6A  D5 000F4         TSTL    LIB$GL_TYPE                              : 0937
                                            7F  12 000F6         BNEQ    12$                                      :
                             50         03  A6  9A 000F8         MOVZBL  3(R6), R0                                : 0947
                             52         04  A6  D0 000FC         MOVL    4(R6), R2                                : 0948
```

```
                62      50      5D   8F  3A 00100          LOCC    #93, R0, (R2)                           ; 0947
                                     02  12 00105          BNEQ    4$
                                     51  D4 00107          CLRL    R1
                        54           51  D0 00109 4$:      MOVL    R1, EXT_START
                                     14  12 0010C          BNEQ    6$                                      ; 0949
                        50      03   A6  9A 0010E          MOVZBL  3(R6), R0                               ; 0950
                62      50           3E  3A 00112          LOCC    #62, R0, (R2)
                                     02  12 00116          BNEQ    5$
                                     51  D4 00118          CLRL    R1
                        54           51  D0 0011A 5$:      MOVL    R1, EXT_START
                                     03  12 0011D          BNEQ    6$                                      ; 0952
                        54           52  D0 0011F          MOVL    R2, EXT_START                           ; 0953
                50      52           54  C3 00122 6$:      SUBL3   EXT_START, R2, R0                       ; 0954
                        51      03   A6  9A 00126          MOVZBL  3(R6), R1
                        50           51  C0 0012A          ADDL2   R1, R0
                64      50           2E  3A 0012D          LOCC    #46, R0, (EXT_START)
                                     02  12 00131          BNEQ    7$
                                     51  D4 00133          CLRL    R1
                        54           51  D0 00135 7$:      MOVL    R1, EXT_START
                50      52           54  C3 00138          SUBL3   EXT_START, R2, R0                       ; 0956
                        51      03   A6  9A 0013C          MOVZBL  3(R6), R1
                        50           51  C0 00140          ADDL2   R1, R0
                64      50           3B  3A 00143          LOCC    #59, R0, (EXT_START)
                                     02  12 00147          BNEQ    8$
                                     51  D4 00149          CLRL    R1
                57      51           54  C3 0014B 8$:      SUBL3   EXT_START, EXT_END, R7                  ; 0963
                                     56  D4 0014F          CLRL    I
                50      56           04  78 00151 9$:      ASHL    #4, I, R0                               ; 0961
                        55  0000GCF 40  9E 00155          MOVAB   LIB_OBJ_DEFEXT[R0], R5
          65    00      64           57  2D 0015B          CMPC5   R7, (EXT_START), #0, (R5), @4(R5)       ; 0963
                                 04  E5      00160
                                     0C  12 00162          BNEQ    11$
                                     55  DD 00164          PUSHL   R5                                      ; 0966
                                 01  A6  9F 00166          PUSHAB  1(I)
                    0000V CF         02  FB 00169          CALLS   #2, SET_LIB_TYPE
                                     07  11 0016E 10$:      BRB     12$                                     ; 0965
                                     56  D6 00170 11$:      INCL    I                                       ; 0958
                        04           56  D1 00172          CMPL    I, #4
                                     DA  1B 00175          BLEQU   9$
                                     6A  D5 00177 12$:      TSTL    LIB$GL_TYPE                             ; 0975
                                     03  12 00179          BNEQ    13$
                        6A           01  D0 0017B          MOVL    #1, LIB$GL_TYPE
                        50           01  D0 0017E 13$:      MOVL    #1, R0                                  ; 0976
                                         04 00181          RET                                             ; 0977

; Routine Size:  386 bytes,    Routine Base:  $CODE$ + 0193
```

```
;  513          0978  1 %SBTTL 'input2file';
;  514          0979  1
;  515          0980  1 ROUTINE input2file (desc) =
;  516          0981  2 BEGIN
;  517          0982  2 !++
;  518          0983  2 ! This routine processes the secondary input file specifiers.
;  519          0984  2 !
;  520          0985  2 !
;  521          0986  2 ! Inputs:
;  522          0987  2 !
;  523          0988  2 !       desc    Address of the descriptor
;  524          0989  2 !
;  525          0990  2 ! Outputs:
;  526          0991  2 !
;  527          0992  2 !       lib$file_scan is called for the file specifier to parse/search and
;  528          0993  2 !       hence get all files included by the specification.  All files searched
;  529          0994  2 !       successfully will be added to the input list.
;  530          0995  2 !
;  531          0996  2 !--
;  532          0997  2
;  533          0998  2 MAP
;  534          0999  2     desc : REF BBLOCK;
;  535          1000  2 OWN
;  536          1001  2     lnamesa : VECTOR [nam$c_maxrss, BYTE],
;  537          1002  2     lnamrsa : VECTOR [nam$c_maxrss, BYTE],
;  538        P 1003  2     lnam : $NAM (
;  539        P 1004  2                 ESA = lnamesa,
;  540        P 1005  2                 ESS = nam$c_maxrss,
;  541        P 1006  2                 RSA = lnamrsa,
;  542          1007  2                 RSS = nam$c_maxrss),
;  543        P 1008  2     lfab : $FAB (
;  544          1009  2                 NAM = lnam),
;  545          1010  2     sticky_context: INITIAL (0),        ! Argument for lib$file_scan.
;  546          1011  2     status;
;  547          1012  2
;  548          1013  2 lib$gl_ctlmsk[lib$v_selective] = cli$present(sd_selective_search);
;  549          1014  2
;  550          1015  2 !
;  551          1016  2 ! Set up the FAB for current input file:
;  552          1017  2 !
;  553          1018  2 lfab[fab$l_fna] = .desc [dsc$a_pointer];
;  554          1019  2 lfab[fab$b_fns] = .desc [dsc$w_length];
;  555          1020  2 lfab[fab$l_dna] = .def_fil_extn [dsc$a_pointer];
;  556          1021  2 lfab[fab$b_dns] = .def_fil_extn [dsc$w_length];
;  557          1022  2
;  558          1023  2 status = lib$file_scan (lfab, filescantruact,          !Find all the files
;  559          1024  2                                filescanflsact, sticky_context);
;  560          1025  2 !
;  561          1026  2 !   lib$file_scan signals any errors, and if no input files are opened, then
;  562          1027  2 !   LIBRARIAN terminates, so no need to retain status.
;  563          1028  2 RETURN true
;  564          1029  1 END;                                                  !Of input2file


                                                        .PSECT  $OWN$,NOEXE,2
```

```
                    001A4 LNAMESA:.BLKB    255
                    002A3         .BLKB    1
                    002A4 LNAMRSA:.BLKB    255
                    003A3         .BLKB    1
                 02 003A4 LNAM:   .BYTE    2
                 60 003A5         .BYTE    96
                 FF 003A6         .BYTE    -1
                 00 003A7         .BYTE    0
          00000000' 003A8         .ADDRESS LNAMRSA
                 00 003AC         .BYTE    0
                 00 003AD         .BYTE    0
                 FF 003AE         .BYTE    -1
                 00 003AF         .BYTE    0
          00000000' 003B0         .ADDRESS LNAMESA
          00000000  003B4         .LONG    0
              0000# 003B8         .WORD    0[8]
              0000# 003C8         .WORD    0[3]
              0000# 003CE         .WORD    0[3]
          00000000  003D4         .LONG    0
          00000000  003D8         .LONG    0
                 00 003DC         .BYTE    0
                 00 003DD         .BYTE    0
                 00 003DE         .BYTE    0
                 00 003DF         .BYTE    0
                 00 003E0         .BYTE    0
                 00 003E1         .BYTE    0
                 00# 003E2        .BYTE    0[2]
          00000000  003E4         .LONG    0
          00000000  003E8         .LONG    0
          00000000  003EC         .LONG    0
          00000000  003F0         .LONG    0
          00000000  003F4         .LONG    0
          00000000  003F8         .LONG    0
          00000000# 003FC         .LONG    0[2]
                 03 00404 LFAB:   .BYTE    3
                 50 00405         .BYTE    80
               0000 00406         .WORD    0
          00000000  00408         .LONG    0
          00000000  0040C         .LONG    0
          00000000  00410         .LONG    0
          00000000  00414         .LONG    0
               0000 00418         .WORD    0
                 02 0041A         .BYTE    2
                 00 0041B         .BYTE    0
          00000000  0041C         .LONG    0
                 00 00420         .BYTE    0
                 00 00421         .BYTE    0
                 00 00422         .BYTE    0
                 02 00423         .BYTE    2
          00000000  00424         .LONG    0
          00000000  00428         .LONG    0
          00000000' 0042C         .ADDRESS LNAM
          00000000  00430         .LONG    0
          00000000  00434         .LONG    0
                 00 00438         .BYTE    0
                 00 00439         .BYTE    0
               0000 0043A         .WORD    0
```

```
                                     00000000 0043C           .LONG    0                                   ;
                                         0000 00440           .WORD    0                                   ;
                                           00 00442           .BYTE    0                                   ;
                                           00 00443           .BYTE    0                                   ;
                                     00000000 00444           .LONG    0                                   ;
                                     00000000 00448           .LONG    0                                   ;
                                         0000 0044C           .WORD    0                                   ;
                                           00 0044E           .BYTE    0                                   ;
                                           00 0044F           .BYTE    0                                   ;
                                     00000000 00450           .LONG    0                                   ;
                                     00000000 00454 STICKY_CONTEXT:                                        ;
                                                              .LONG    0                                   ;
                                              00458 STATUS:   .BLKB    4


                                                              .PSECT   $CODE$,NOWRT,2

                                         0004 00000 INPUT2FILE:
                                                              .WORD    Save R2                             ; 0980
                          52      0000'  CF   9E 00002        MOVAB    LFAB+44, R2                         ;
                                  0000'  CF   9F 00007        PUSHAB   SD_SELECTIVE_SEARCH                 ; 1013
                  0000G   CF             01   FB 0000B        CALLS    #1, CLI$PRESENT                     ;
   0000G  CF         01   02             50   F0 00010        INSV     R0, #2, #1, LIB$GL_CTLMSK+2         ;
                          50        04   AC   D0 00017        MOVL     DESC, R0                            ; 1018
                          62        04   A0   D0 0001B        MOVL     4(R0), LFAB+44                      ;
                     08   A2             60   90 0001F        MOVB     (R0), LFAB+52                       ; 1019
                     04   A2      FCF0   C2   D0 00023        MOVL     DEF_FIL_EXTN+4, LFAB+48             ; 1020
                     09   A2      FCEC   C2   90 00029        MOVB     DEF_FIL_EXTN, LFAB+53               ; 1021
                                    24   A2   9F 0002F        PUSHAB   STICKY_CONTEXT                      ; 1023
                                  0000V  CF   9F 00032        PUSHAB   FILESCANFLSACT                      ;
                                  0000V  CF   9F 00036        PUSHAB   FILESCANTRUACT                      ;
                                    D4   A2   9F 0003A        PUSHAB   LFAB                                ;
              00000000G   00             04   FB 0003D        CALLS    #4, LIB$FILE_SCAN                   ;
                     28   A2             50   D0 00044        MOVL     R0, STATUS                          ;
                          50             01   D0 00048        MOVL     #1, R0                              ; 1028
                                              04 0004B        RET                                          ; 1029

; Routine Size:  76 bytes,    Routine Base:  $CODE$ + 0315
```

```
;  566     1030  1 %SBTTL 'filescantruact';
;  567     1031  1
;  568     1032  1 ROUTINE filescantruact (fab) =
;  569     1033  2 BEGIN
;  570     1034  2 !++
;  571     1035  2 ! This routine is called by lib$file_scan when a successful $search has
;  572     1036  2 ! been done.
;  573     1037  2 !
;  574     1038  2 !
;  575     1039  2 ! Inputs:
;  576     1040  2 !
;  577     1041  2 !       fab     Address of the fab
;  578     1042  2 !
;  579     1043  2 ! Outputs:
;  580     1044  2 !
;  581     1045  2 !       a file descriptor block is allocated and the information is copied
;  582     1046  2 !       into it so the file can be opened later.
;  583     1047  2 !
;  584     1048  2 !--
;  585     1049  2
;  586     1050  2 MAP
;  587     1051  2     fab : REF BBLOCK;
;  588     1052  2 LOCAL
;  589     1053  2     module_string,
;  590     1054  2     saversa,
;  591     1055  2     namblk : REF BBLOCK,
;  592     1056  2     newnamdesc : REF BBLOCK,
;  593     1057  2     newfdb : REF BBLOCK;
;  594     1058  2
;  595     1059  2 BIND
;  596     1060  2     scaname = .fab [fab$l_nam] : BBLOCK;         !NAM block of winner
;  597     1061  2
;  598     1062  2 allocate_fdb (newfdb);                          !Get a new fdb
;  599     1063  2 newnamdesc = newfdb [fdb$l_namdesc];            !Point to filename descriptor
;  600     1064  2 lastfdb [fdb$l_nxtfdb] = .newfdb;               !Link into the list
;  601     1065  2 prevfdb = .lastfdb;
;  602     1066  2 lastfdb = .newfdb;                             !Set new last
;  603     1067  2 namblk = newfdb [fdb$t_nam];                   !Point to new nam block
;  604     1068  2 saversa = .namblk [nam$l_rsa];                 !Save over CH$MOVE
;  605     1069  2 CH$MOVE (nam$c_bln, scaname, .namblk);  !Copy NAM
;  606     1070  2 namblk [nam$l_esa] = namblk [nam$l_rsa] = .saversa; !Set ESA to RSA
;  607     1071  2 namblk [nam$b_rsl] = .scaname [nam$b_rsl];
;  608     1072  2 namblk [nam$b_esl] = .scaname [nam$b_rsl];
;  609     1073  2 getfilnamdesc (.fab, .newnamdesc);              !Get string descriptor for file spec.
;  610     1074  2 CH$MOVE (.namblk [nam$b_rsl], .scaname [nam$l_rsa], .namblk [nam$l_rsa]);
;  611     1075  2 if .modulenamedesc[dsc$w_length] gtr 0
;  612     1076  2 then
;  613     1077  3     begin
;  614     1078  3     lib_get_mem(.modulenamedesc[dsc$w_length], module_string);
;  615     1079  3     ch$move(.modulenamedesc[dsc$w_length], .modulenamedesc[dsc$a_pointer], .module_string);
;  616     1080  3     newfdb[fdb$l_modnam] = .modulenamedesc[dsc$w_length];
;  617     1081  3     newfdb[fdb$l_modnam] + 4 = .module_string;
;  618     1082  2     end;
;  619     1083  2
;  620     1084  2 RETURN true
;  621     1085  1 END;                                                    !Of filescantruact
```

```
                                    0FFC 00000 FILESCANTRUACT:
                                               .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11           ; 1032
                5E            08    C2 00002   SUBL2    #8, SP                                         ;
                5B      04    AC    D0 00005   MOVL     FAB, R11                                       ; 1060
                58      28    AB    D0 00009   MOVL     40(R11), R8                                    ;
                              5E    DD 0000D   PUSHL    SP                                             ; 1062
        0000V   CF            01    FB 0000F   CALLS    #1, ALLOCATE_FDB                               ;
                59            6E    D0 00014   MOVL     NEWFDB, R9                                     ; 1063
                57      10    A9    9E 00017   MOVAB    16(R9), NEWNAMDESC                             ;
        0000'   DF            59    D0 0001B   MOVL     R9, @LASTFDB                                   ; 1064
        0000'   CF    0000'   CF    D0 00020   MOVL     LASTFDB, PREVFDB                               ; 1065
        0000'   CF            59    D0 00027   MOVL     R9, LASTFDB                                    ; 1066
                56      40    A9    9E 0002C   MOVAB    64(R9), NAMBLK                                 ; 1067
                5A      04    A6    D0 00030   MOVL     4(NAMBLK), SAVERSA                             ; 1068
     66         68    0060    8F    28 00034   MOVC3    #96, (R8), (NAMBLK)                            ; 1069
           04   A6            5A    D0 0003A   MOVL     SAVERSA, 4(NAMBLK)                             ; 1070
           0C   A6            5A    D0 0003E   MOVL     SAVERSA, 12(NAMBLK)                            ;
           03   A6      03    A8    90 00042   MOVB     3(R8), 3(NAMBLK)                               ; 1071
           0B   A6      03    A8    90 00047   MOVB     3(R8), 11(NAMBLK)                              ; 1072
                              57    DD 0004C   PUSHL    NEWNAMDESC                                     ; 1073
                              5B    DD 0004E   PUSHL    R11                                            ;
        0000G   CF            02    FB 00050   CALLS    #2, GETFILNAMDESC                              ;
                50      03    A6    9A 00055   MOVZBL   3(NAMBLK), R0                                  ; 1074
04   B6    04   B8            50    28 00059   MOVC3    R0, @4(R8), @4(NAMBLK)                         ;
                50    0000'   CF    3C 0005F   MOVZWL   MODULENAMEDESC, R0                             ; 1075
                              1E    15 00064   BLEQ     1$                                             ;
                        04    AE    9F 00066   PUSHAB   MODULE_STRING                                  ; 1078
                              50    DD 00069   PUSHL    R0                                             ;
        0000G   CF            02    FB 0006B   CALLS    #2, LIB_GET_MEM                                ;
04   BE 0000'   DF    0000'   CF    28 00070   MOVC3    MODULENAMEDESC, @MODULENAMEDESC+4, -           ; 1079
                                                        @MODULE_STRING                                 ;
           18   A9    0000'   CF    3C 00079   MOVZWL   MODULENAMEDESC, 24(R9)                         ; 1080
           1C   A9      04    AE    D0 0007F   MOVL     MODULE_STRING, 28(R9)                          ; 1081
                50            01    D0 00084 1$: MOVL   #1, R0                                         ; 1084
                                    04 00087   RET                                                     ; 1085

; Routine Size:  136 bytes,    Routine Base:  $CODE$ + 0361
```

```
1 %SBTTL 'filescanflsact';
1
1 ROUTINE filescanflsact (fab) =
2 BEGIN
2 !++
2 ! This routine is called by lib$file_scan when an unsuccessful $search has
2 ! been done. Ignore it after issuing an error message
2 !
2 ! Inputs:
2 !
2 !       fab     Address of the fab
2 !
2 ! Outputs:
2 !
2 !       An error message is issued.
2 !
2 !--
2
2 MAP
2     fab : REF BBLOCK;
2
2 LOCAL
2     filedesc : BBLOCK [dsc$c_s_bln];                     !String desc. for filename
2
2 !
2 ! Get the filename string descriptor set up
2 !
2 getfilnamdesc (.fab, filedesc);
2 SIGNAL (lib$_openin, 1, filedesc, .fab [fab$l_sts], .fab [fab$l_stv]);
2 filesnotfound = .filesnotfound + 1;
2
2 RETURN true
1 END;                                                     !Of filescanflsact
```

```
                                   0004 00000 FILESCANFLSACT:
                                                        .WORD   Save R2                     ; 1088
               5E            08 C2 00002                SUBL2   #8, SP
                             5E DD 00005                PUSHL   SP                          ; 1113
               52        04  AC D0 00007                MOVL    FAB, R2
                             52 DD 0000B                PUSHL   R2
        0000G  CF            02 FB 0000D                CALLS   #2, GETFILNAMDESC
               7E        08  A2 7D 00012                MOVQ    8(R2), -(SP)                ; 1114
                         08  AE 9F 00016                PUSHAB  FILEDESC
                             01 DD 00019                PUSHL   #1
                   00861098  8F DD 0001B                PUSHL   #8786072
    00000000G  00            05 FB 00021                CALLS   #5, LIB$SIGNAL
                      0000'  CF D6 00028                INCL    FILESNOTFOUND               ; 1115
               50            01 D0 0002C                MOVL    #1, R0                      ; 1117
                                04 0002F                RET                                 ; 1118

; Routine Size:  48 bytes,    Routine Base:  $CODE$ + 03E9
```

```
;  657     1119  1 %SBTTL 'outputfile & setlistfile';
;  658     1120  1
;  659     1121  1 ROUTINE outputfile =
;  660     1122  2 BEGIN
;  661     1123  2 !
;  662     1124  2 ! This routine processes output files.
;  663     1125  2 !
;  664     1126  2 LOCAL
;  665     1127  2     filenamestring;
;  666     1128  2
;  667     1129  2 allocate_fdb (lib$gl_outfdb);                          !Allocate FDB for output
;  668     1130  2 lib_get_mem(.token_desc[dsc$w_length], filenamestring);
;  669     1131  2 ch$move (.token_desc[dsc$w_length], .token_desc[dsc$a_pointer], .filenamestring);
;  670     1132  2 lib$gl_outfdb[fdb$l_namdesc] = .token_desc[dsc$w_length];
;  671     1133  2 lib$gl_outfdb[fdb$l_namdesc] + 4 = .filenamestring;
;  672     1134  2 CH$MOVE (dsc$c_s_bln, def_fil_extn, lib$gl_outfdb [fdb$l_defext]); !set default extension
;  673     1135  2 RETURN true                                            !That's all
;  674     1136  1 END;



                                           007C 00000 OUTPUTFILE:
                                                          .WORD    Save R2,R3,R4,R5,R6                     ; 1121
                        56      0000'  CF   9E 00002      MOVAB    TOKEN_DESC, R6
                        5E             04   C2 00007      SUBL2    #4, SP
                                0000G  CF   9F 0000A      PUSHAB   LIB$GL_OUTFDB                           ; 1129
                  0000V CF             01   FB 0000E      CALLS    #1, ALLOCATE_FDB
                                       5E   DD 00013      PUSHL    SP                                      ; 1130
                        7E             66   3C 00015      MOVZWL   TOKEN_DESC, -(SP)
                  0000G CF             02   FB 00018      CALLS    #2, LIB_GET_MEM
        00   BE    04   B6             66   28 0001D      MOVC3    TOKEN_DESC, @TOKEN_DESC+4, @FILENAMESTRING ; 1131
                        50      0000G  CF   D0 00023      MOVL     LIB$GL_OUTFDB, R0                       ; 1132
                   10   A0             66   3C 00028      MOVZWL   TOKEN_DESC, 16(R0)
                   14   A0             6E   D0 0002C      MOVL     FILENAMESTRING, 20(R0)                  ; 1133
        08   A0    D8   A6             08   28 00030      MOVC3    #8, DEF_FIL_EXTN, 8(R0)                 ; 1134
                        50             01   D0 00036      MOVL     #1, R0                                  ; 1135
                                            04 00039      RET                                              ; 1136

; Routine Size:  58 bytes,    Routine Base:  $CODE$ + 0419

;  675     1137  1 ROUTINE setlistfile  =
;  676     1138  2 BEGIN
;  677     1139  2 !
;  678     1140  2 ! This routine is called when the /LIST qualifier is parsed
;  679     1141  2 !
;  680     1142  2
;  681     1143  2 allocate_fdb (lib$gl_lisfdb);                  !Allocate FDB for list file
;  682     1144  2 cli$get_value(sd_list, token_desc);
;  683     1145  2 if .token_desc[dsc$w_length] NEQ 0             !If /LIST=string
;  684     1146  2 THEN
;  685     1147  3     BEGIN
;  686     1148  3     LOCAL
;  687     1149  3         filenamestring;
;  688     1150  3     lib_get_mem(.token_desc[dsc$w_length], filenamestring);
```

```
;  689      1151  3        ch$move (.token_desc[dsc$w_length], .token_desc[dsc$a_pointer], .filenamestring);
;  690      1152  3        lib$gl_lisfdb[fdb$l_namdesc] = .token_desc[dsc$w_length];
;  691      1153  3        lib$gl_lisfdb[fdb$l_namdesc] + 4 = .filenamestring;
;  692      1154  3        END
;  693      1155  2    ELSE
;  694      1156  2        CH$MOVE (dsc$c_s_bln, sysoutputdesc, lib$gl_lisfdb [fdb$l_namdesc]);
;  695      1157  2    RETURN true
;  696      1158  1    END;


                                          00FC 00000 SETLISTFILE:
                                                           .WORD   Save R2,R3,R4,R5,R6,R7                        ; 1137
                         57      0000G  CF   9E 00002      MOVAB   LIB$GL_LISFDB, R7                             ;
                         56      0000'  CF   9E 00007      MOVAB   TOKEN_DESC, R6                                ;
                         5E             04   C2 0000C      SUBL2   #4, SP                                        ;
                                        57   DD 0000F      PUSHL   R7                                            ; 1143
                 0000V   CF             01   FB 00011      CALLS   #1, ALLOCATE_FDB                              ;
                                        56   DD 00016      PUSHL   R6                                            ; 1144
                                 0000'  CF   9F 00018      PUSHAB  SD_LIST                                       ;
                 0000G   CF             02   FB 0001C      CALLS   #2, CLI$GET_VALUE                             ;
                         50             66   3C 00021      MOVZWL  TOKEN_DESC, R0                                ; 1145
                                        1C   13 00024      BEQL    1$                                            ;
                                 4001   8F   BB 00026      PUSHR   #^M<R0,SP>                                    ; 1150
                 0000G   CF             02   FB 0002A      CALLS   #2, LIB_GET_MEM                               ;
      00   BE       04   B6             66   28 0002F      MOVC3   TOKEN_DESC, @TOKEN_DESC+4, @FILENAMESTRING    ; 1151
                         50             67   D0 00035      MOVL    LIB$GL_LISFDB, R0                             ; 1152
                 10      A0             66   3C 00038      MOVZWL  TOKEN_DESC, 16(R0)                            ;
                 14      A0             6E   D0 0003C      MOVL    FILENAMESTRING, 20(R0)                        ; 1153
                                        09   11 00040      BRB     2$                                            ; 1145
                         50             67   D0 00042 1$:  MOVL    LIB$GL_LISFDB, R0                             ; 1156
      10   A0       E8   A6             08   28 00045      MOVC3   #8, SYSOUTPUTDESC, 16(R0)                     ;
                         50             01   D0 0004B 2$:  MOVL    #1, R0                                        ; 1157
                                             04 0004E      RET                                                   ; 1158

; Routine Size:  79 bytes,    Routine Base:  $CODE$ + 0453


;  697      1159  1
;  698      1160  1    %SBTTL 'setlistwidth';
;  699      1161  1    ROUTINE setlistwidth  =
;  700      1162  2    BEGIN
;  701      1163  2    !
;  702      1164  2    ! set list for /list
;  703      1165  2    !
;  704      1166  2    cli$get_value(sd_width, token_desc);
;  705      1167  2    lib$cvt_dtb(.token_desc[dsc$w_length], .token_desc[dsc$a_pointer], lib$gl_listwid);
;  706      1168  2    RETURN true
;  707      1169  1    END;


                                          0000 00000 SETLISTWIDTH:
```

```
                                                               .WORD    Save nothing                          ; 1161
                                    0000'  CF  9F  00002       PUSHAB   TOKEN_DESC                            ; 1166
                                    0000'  CF  9F  00006       PUSHAB   SD_WIDTH                              ;
                        0000G  CF          02  FB  0000A       CALLS    #2, CLI$GET_VALUE                     ;
                                    0000G  CF  9F  0000F       PUSHAB   LIB$GL_LISTWID                        ; 1167
                                    0000'  CF  DD  00013       PUSHL    TOKEN_DESC+4                          ;
                               7E   0000'  CF  3C  00017       MOVZWL   TOKEN_DESC, -(SP)                     ;
                    00000000G  00          03  FB  0001C       CALLS    #3, LIB$CVT_DTB                       ;
                               50          01  D0  00023       MOVL     #1, R0                                ; 1168
                                               04  00026       RET                                            ; 1169

; Routine Size:  39 bytes,    Routine Base:  $CODE$ + 04A2


;   708          1170  1
;   709          1171  1 %SBTTL 'extractfile';
;   710          1172  1 ROUTINE extractfile (desc) =
;   711          1173  2 BEGIN
;   712          1174  2 !
;   713          1175  2 ! This routine is called when the /EXTRACT qualifier is parsed
;   714          1176  2 !
;   715          1177  2 get_name_list (lib$gl_modxtrl, sd_extract);      !Get the list of modules to extract
;   716          1178  2 RETURN true
;   717          1179  1 END;



                                            0000  00000 EXTRACTFILE:
                                                               .WORD    Save nothing                          ; 1172
                                    0000'  CF  9F  00002       PUSHAB   SD_EXTRACT                            ; 1177
                                    0000G  CF  9F  00006       PUSHAB   LIB$GL_MODXTRL                        ;
                        0000V  CF          02  FB  0000A       CALLS    #2, GET_NAME_LIST                     ;
                               50          01  D0  0000F       MOVL     #1, R0                                ; 1178
                                               04  00012       RET                                            ; 1179

; Routine Size:  19 bytes,    Routine Base:  $CODE$ + 04C9


;   718          1180  1
;   719          1181  1 %SBTTL 'createlib';
;   720          1182  1 ROUTINE createlib =
;   721          1183  2 BEGIN
;   722          1184  2 !
;   723          1185  2 ! Called by CLI when the /CREATE qualifier is seen. Scan the create options
;   724          1186  2 !
;   725          1187  2 scan_options (sd_create, create_states, create_keys, lib$al_creopts, lib$gl_cre8flags);
;   726          1188  2 RETURN true
;   727          1189  1 END;



                                            0000  00000 CREATELIB:
                                                               .WORD    Save nothing                          ; 1182
```

```
                                 0000G  CF  9F 00002         PUSHAB  LIB$GL_CRE8FLAGS                        ; 1187
                                 0000G  CF  9F 00006         PUSHAB  LIB$AL_CREOPTS                          ;
                                 0000'  CF  9F 0000A         PUSHAB  CREATE_KEYS                             ;
                                 0000'  CF  9F 0000E         PUSHAB  CREATE_STATES                           ;
                                 0000'  CF  9F 00012         PUSHAB  SD_CREATE                               ;
                          0000V  CF          05 FB 00016      CALLS   #5, SCAN_OPTIONS                        ;
                             50              01 D0 0001B      MOVL    #1, R0                                  ; 1188
                                             04 0001E         RET                                             ; 1189

; Routine Size:  31 bytes,    Routine Base:  $CODE$ + 04DC

;  728          1190  1
;  729          1191  1 %SBTTL 'compressfile';
;  730          1192  1 ROUTINE compressfile  =
;  731          1193  2 BEGIN
;  732          1194  2 !
;  733          1195  2 ! This routine is called when the /COMPRESS qualifier is parsed
;  734          1196  2 !
;  735          1197  2 scan_options (sd_compress, compress_states, compress_keys, lib$al_creopts, lib$gl_cre8flags);
;  736          1198  2 IF .lib$gl_cre8flags [lib$c_opt_keep] THEN lib$gl_ctlmsk [lib$v_keep] = true;
;  737          1199  2 IF .lib$gl_cre8flags [lib$c_opt_dcx] THEN lib$gl_ctlmsk [lib$v_dcx] = true;
;  738          1200  2 RETURN true
;  739          1201  1 END;


                                 0004 00000 COMPRESSFILE:
                                                 .WORD   Save R2                                         ; 1192
                             52  0000G  CF  9E 00002         MOVAB   LIB$GL_CRE8FLAGS, R2                    ;
                                 52      DD 00007         PUSHL   R2                                      ; 1197
                                 0000G  CF  9F 00009         PUSHAB  LIB$AL_CREOPTS                          ;
                                 0000'  CF  9F 0000D         PUSHAB  COMPRESS_KEYS                           ;
                                 0000'  CF  9F 00011         PUSHAB  COMPRESS_STATES                         ;
                                 0000'  CF  9F 00015         PUSHAB  SD_COMPRESS                             ;
                          0000V  CF          05 FB 00019      CALLS   #5, SCAN_OPTIONS                        ;
                      06              62     06 E1 0001E      BBC     #6, LIB$GL_CRE8FLAGS, 1$                ; 1198
                          0000G  CF      80  8F 88 00022      BISB2   #128, LIB$GL_CTLMSK+3                   ;
                                             62 95 00028 1$:  TSTB    LIB$GL_CRE8FLAGS                        ; 1199
                                             05 18 0002A      BGEQ    2$                                      ;
                          0000G  CF          04 88 0002C      BISB2   #4, LIB$GL_CTLMSK+4                     ;
                             50              01 D0 00031 2$:  MOVL    #1, R0                                  ; 1200
                                             04 00034         RET                                             ; 1201

; Routine Size:  53 bytes,    Routine Base:  $CODE$ + 04FB

;  740          1202  1
;  741          1203  1 %SBTTL 'datareduce';
;  742          1204  1 ROUTINE datareduce =
;  743          1205  2 BEGIN
;  744          1206  2 !
;  745          1207  2 ! This routine is called when the /DATA qualifier is parsed
;  746          1208  2 !
;  747          1209  2 scan_options (sd_data, data_states, data_keys, lib$al_creopts, lib$gl_cre8flags);
```

```
;  748          1210  2 IF .lib$gl_cre8flags [lib$c_opt_dcx] THEN lib$gl_ctlmsk [lib$v_dcx] = true;
;  749          1211  2 RETURN true
;  750          1212  1 END;


                                        0000 00000 DATAREDUCE:
                                                   .WORD   Save nothing                    ; 1204
                                0000G   CF   9F 00002      PUSHAB  LIB$GL_CRE8FLAGS        ; 1209
                                0000G   CF   9F 00006      PUSHAB  LIB$AL_CREOPTS          :
                                0000'   CF   9F 0000A      PUSHAB  DATA_KEYS               :
                                0000'   CF   9F 0000E      PUSHAB  DATA_STATES             :
                                0000'   CF   9F 00012      PUSHAB  SD_DATA                 :
                          0000V CF          05   FB 00016      CALLS   #5, SCAN_OPTIONS      :
                                0000G   CF   95   0001B      TSTB    LIB$GL_CRE8FLAGS      ; 1210
                                        05   18   0001F      BGEQ    1$                    :
                          0000G CF          04   88 00021      BISB2   #4, LIB$GL_CTLMSK+4   :
                                50          01   D0 00026 1$:  MOVL    #1, R0              ; 1211
                                            04   00029        RET                           ; 1212

; Routine Size:  42 bytes,    Routine Base:  $CODE$ + 0530

;  751          1213  1
;  752          1214  1 %SBTTL 'removesymbols';
;  753          1215  1 ROUTINE removesymbols =
;  754          1216  2 BEGIN
;  755          1217  2 !
;  756          1218  2 ! This routine is called when the /REMOVE qualifier is parsed
;  757          1219  2 !
;  758          1220  2 get_name_list (lib$gl_objsyrml,sd_remove);      !Get the list of symbols to remove
;  759          1221  2 set_lib_type (lib$s_object, lib_obj_defext);          !Set object type
;  760          1222  2 RETURN true
;  761          1223  1 END;


                                        0000 00000 REMOVFSYMBOLS:
                                                   .WORD   Save nothing                    ; 1215
                                0000'   CF   9F 00002      PUSHAB  SD_REMOVE               ; 1220
                                0000G   CF   9F 00006      PUSHAB  LIB$GL_OBJSYRML         :
                          0000V CF          02   FB 0000A      CALLS   #2, GET_NAME_LIST     :
                                0000G   CF   9F   0000F      PUSHAB  LIB_OBJ_DEFEXT          ; 1221
                                        01   DD 00013        PUSHL   #1                    :
                          0000V CF          02   FB 00015      CALLS   #2, SET_LIB_TYPE      :
                                50          01   D0 0001A      MOVL    #1, R0              ; 1222
                                            04   0001D        RET                           ; 1223

; Routine Size:  30 bytes,    Routine Base:  $CODE$ + 055A

;  762          1224  1
;  763          1225  1 %SBTTL 'deletemodules';
```

```
;  764          1226  1 ROUTINE deletemodules  =
;  765          1227  2 BEGIN
;  766          1228  2 !
;  767          1229  2 ! This routine is called when the /DELETE qualifier is parsed
;  768          1230  2 !
;  769          1231  2 get_name_list (lib$gl_delmodl, sd_delete);      !Get module names to delete
;  770          1232  2 RETURN true
;  771          1233  1 END;


                                0000 00000 DELETEMODULES:
                                                .WORD   Save nothing                    ; 1226
                                0000'  CF  9F 00002     PUSHAB  SD_DELETE               ; 1231
                                0000G  CF  9F 00006     PUSHAB  LIB$GL_DELMODL          ;
                          0000V  CF         02  FB 0000A  CALLS   #2, GET_NAME_LIST     ;
                                 50         01  D0 0000F  MOVL    #1, R0               ; 1232
                                            04 00012    RET                             ; 1233

; Routine Size:  19 bytes,    Routine Base:  $CODE$ + 0578
```

```
;  773          1234  1 %SBTTL 'listonlymods';
;  774          1235  1 ROUTINE listonlymods  =
;  775          1236  2 BEGIN
;  776          1237  2 !
;  777          1238  2 ! This routine is called when the /ONLY qualifier is parsed
;  778          1239  2 !
;  779          1240  2 get_name_list (lib$gl_modlisl, sd_only);
;  780          1241  2 RETURN true
;  781          1242  1 END;


                                        0000 00000 LISTONLYMODS:
                                                   .WORD   Save nothing                    ; 1235
                                0000'   CF  9F 00002       PUSHAB  SD_ONLY                 ; 1240
                                0000G   CF  9F 00006       PUSHAB  LIB$GL_MODLISL          ;
                        0000V   CF         02  FB 0000A    CALLS   #2, GET_NAME_LIST       ;
                                50         01  D0 0000F    MOVL    #1, R0                  ; 1241
                                           04  00012       RET                             ; 1242

; Routine Size:  19 bytes,    Routine Base:  $CODE$ + 058B

;  782          1243  1
;  783          1244  1 %SBTTL 'setmodulename';
;  784          1245  1
;  785          1246  1 ROUTINE setmodulename  =
;  786          1247  2 BEGIN
;  787          1248  2 !
;  788          1249  2 ! This routine processes the /MODULES qualifier.  It is an input-file specific
;  789          1250  2 ! qualifier.
;  790          1251  2 !
;  791          1252  2
;  792          1253  2 cli$get_value(sd_modules,modulenamedesc);
;  793          1254  2 IF .modulenamedesc[dsc$w_length] EQL 0
;  794          1255  2 THEN SIGNAL_STOP (lib$_novalue);
;  795          1256  2 RETURN true
;  796          1257  1 END;


                                        0000 00000 SETMODULENAME:
                                                   .WORD   Save nothing                    ; 1246
                                0000'   CF  9F 00002       PUSHAB  MODULENAMEDESC          ; 1253
                                0000'   CF  9F 00006       PUSHAB  SD_MODULES              ;
                        0000G   CF         02  FB 0000A    CALLS   #2, CLI$GET_VALUE       ;
                                0000'   CF  B5 0000F       TSTW    MODULENAMEDESC          ; 1254
                                           0D  12 00013    BNEQ    1$                      ;
                                00861104   8F  DD 00015    PUSHL   #8786180                ; 1255
                        00000000G  00      01  FB 0001B    CALLS   #1, LIB$STOP            ;
                                50         01  D0 00022 1$: MOVL    #1, R0                  ; 1256
                                           04  00025       RET                             ; 1257
```

; Routine Size: 38 bytes,   Routine Base: $CODE$ + 059E

```
;   798          1258  1 %SBTTL 'scan_options';
;   799          1259  1
;   800          1260  1 ROUTINE scan_options ( qualifier_desc, tp_states, tp_keys, resultvector, resultflags) =
;   801          1261  2 BEGIN
;   802          1262  2 !
;   803          1263  2 ! This routine recalls CLI to get the options for
;   804          1264  2 ! /CREATE=(BLOCKS=123,GLOBALS=456,MODULES=321,HISTORY=654,VERSION:2) and
;   805          1265  2 ! /COMPRESS = (BLO=123, GLOB=456, MOD=321, HIST=654, VERSION:2, KEEP)
;   806          1266  2 !
;   807          1267  2 MAP
;   808          1268  2     qualifier_desc : ref bblock,
;   809          1269  2     resultvector : REF VECTOR,
;   810          1270  2     resultflags : REF BITVECTOR;
;   811          1271  2
;   812          1272  2 LOCAL
;   813          1273  2     ptr,
;   814          1274  2     value_desc : bblock[dsc$c_s_bln],
;   815          1275  2     key_desc : bblock[dsc$c_s_bln],
;   816          1276  2     status;
;   817          1277  2
;   818          1278  2 ! Recall CLI for the option name, then recall it again to get the string
;   819          1279  2 ! containing the value. convert the decimal value to binary.
;   820          1280  2 !
;   821          1281  2 WHILE cli$get_value(.qualifier_desc, token_desc)  DO            !While more to do
;   822          1282  3     BEGIN
;   823          1283  3     ch$move(dsc$c_s_bln, token_desc, key_desc);
;   824          1284  3     IF ch$fail( ptr=ch$find_ch(.token_desc[dsc$w_length],.token_desc[dsc$a_pointer],%c'='))
;   825          1285  3     THEN
;   826          1286  3         ptr = ch$find_ch(.token_desc[dsc$w_length],.token_desc[dsc$a_pointer],%c':');
;   827          1287  3     IF .ptr eql 0
;   828          1288  3     then
;   829          1289  3         value_desc[dsc$w_length] = 0
;   830          1290  3     else
;   831          1291  4         begin
;   832          1292  4         value_desc[dsc$a_pointer] = .ptr + 1 ;
;   833          1293  4         value_desc[dsc$w_length] = .token_desc[dsc$w_length] - (.ptr - .token_desc[dsc$a_pointer]) - 1;
;   834          1294  4         key_desc[dsc$w_length] = .token_desc[dsc$w_length] - .value_desc[dsc$w_length] - 1;
;   835          1295  3         end;
;   836          1296  3     !
;   837          1297  3     !   call tparse
;   838          1298  3     !
;   839          1299  3     lib$gl_tpindex = 0;
;   840          1300  3     lib$gl_valreq = 0;
;   841          1301  3     status = call_tparse (key_desc, .tp_states, .tp_keys);
;   842          1302  3     IF .lib$gl_valreq
;   843          1303  3     THEN
;   844          1304  4         begin
;   845          1305  4         if .value_desc[dsc$w_length] eql 0
;   846          1306  4         then
;   847          1307  4             SIGNAL_STOP ( lib$_novalue, 1, token_desc [dsc$w_length])
;   848          1308  4         ELSE
;   849          1309  4             IF NOT lib$cvt_dtb(.value_desc[dsc$w_length], .value_desc[dsc$a_pointer],
;   850          1310  4                                resultvector[.lib$gl_tpindex])
;   851          1311  4             THEN SIGNAL_STOP (lib$_badkey, 1, value_desc [dsc$w_length]);
;   852          1312  4         lib$gl_cre8flags = .lib$gl_cre8flags OR 1^.lib$gl_tpindex;
;   853          1313  3         END;
;   854          1314  2 END;
```

```
;  855          1315  2 RETURN .status;
;  856          1316  1 END;                                    !OF scan_options



                                     03FC 00000 SCAN_OPTIONS:
                                                       .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9                         ; 1260
                 59      0000'  CF  9E 00002           MOVAB    LIB$GL_TPINDEX, R9                                   ;
                 58      0000'  CF  9E 00007           MOVAB    TOKEN_DESC, R8                                       ;
                 5E             10  C2 0000C           SUBL2    #16, SP                                              ;
                                58  DD 0000F 1$:       PUSHL    R8                                                   ; 1281
                           04   AC  DD 00011           PUSHL    QUALIFIER_DESC                                       ;
         0000G   CF             02  FB 00014           CALLS    #2, CLI$GET_VALUE                                    ;
                 03             50  E8 00019           BLBS     R0, 2$                                               ;
                              00A3  31 0001C           BRW      10$                                                  ;
    6E           68             08  28 0001F 2$:       MOVC3    #8, TOKEN_DESC, KEY_DESC                             ; 1283
                 52             68  3C 00023           MOVZWL   TOKEN_DESC, R2                                       ; 1284
                 53        04   A8  D0 00026           MOVL     TOKEN_DESC+4, R3                                     ;
    63           52             3D  3A 0002A           LOCC     #61, R2, (R3)                                        ;
                                02  12 0002E           BNEQ     3$                                                   ;
                                51  D4 00030           CLRL     R1                                                   ;
                 56             51  D0 00032 3$:       MOVL     R1, PTR                                              ;
                                12  12 00035           BNEQ     5$                                                   ;
    63           52             3A  3A 00037           LOCC     #58, R2, (R3)                                        ; 1286
                                02  12 0003B           BNEQ     4$                                                   ;
                                51  D4 0003D           CLRL     R1                                                   ;
                 56             51  D0 0003F 4$:       MOVL     R1, PTR                                              ;
                                05  12 00042           BNEQ     5$                                                   ; 1287
                           08   AE  B4 00044           CLRW     VALUE_DESC                                           ; 1289
                                1D  11 00047           BRB      6$                                                   ;
            0C   AE        01   A6  9E 00049 5$:       MOVAB    1(R6), VALUE_DESC+4                                  ; 1292
    50           53             56  C3 0004E           SUBL3    PTR, R3, R0                                          ; 1293
                 51        FF A042  9E 00052           MOVAB    -1(R0)[R2], R1                                       ;
            08   AE             51  B0 00057           MOVW     R1, VALUE_DESC                                       ;
                 50        08   AE  3C 0005B           MOVZWL   VALUE_DESC, R0                                       ; 1294
                 52             50  C2 0005F           SUBL2    R0, R2                                               ;
    6E           52             01  A3 00062           SUBW3    #1, R2, KEY_DESC                                     ;
                                69  7C 00066 6$:       CLRQ     LIB$GL_TPINDEX                                       ; 1299
                 7E        08   AC  7D 00068           MOVQ     TP_STATES, -(SP)                                     ; 1301
                           08   AE  9F 0006C           PUSHAB   KEY_DESC                                             ;
         0000V   CF             03  FB 0006F           CALLS    #3, CALL_TPARSE                                      ;
                 57             50  D0 00074           MOVL     R0, STATUS                                           ;
                 94        04   A9  E9 00077           BLBC     LIB$GL_VALREQ, 1$                                    ; 1302
                           08   AE  B5 0007B           TSTW     VALUE_DESC                                           ; 1305
                                0C  12 0007E           BNEQ     7$                                                   ;
                                58  DD 00080           PUSHL    R8                                                   ; 1307
                                01  DD 00082           PUSHL    #1                                                   ;
                    00861104    8F  DD 00084           PUSHL    #8786180                                             ;
                                23  11 0008A           BRB      8$                                                   ;
                 50             69  D0 0008C 7$:       MOVL     LIB$GL_TPINDEX, R0                                   ; 1310
                           10 BC40  DF 0008F           PUSHAL   @RESULTVECTOR[R0]                                    ;
                           10   AE  DD 00093           PUSHL    VALUE_DESC+4                                         ;
                 7E        10   AE  3C 00096           MOVZWL   VALUE_DESC, -(SP)                                    ;
     00000000G   00             03  FB 0009A           CALLS    #3, LIB$CVT_DTB                                      ;
                 12             50  E8 000A1           BLBS     R0, 9$                                               ;
```

```
                              08   AE  9F 000A4        PUSHAB  VALUE_DESC                      ; 1311
                                   01  DD 000A7        PUSHL   #1                              ;
                      0086110C     8F  DD 000A9        PUSHL   #8786188                        ;
          00000000G  00            03  FB 000AF  8$:   CALLS   #3, LIB$STOP                    ;
       50            01            69  78 000B6  9$:   ASHL    LIB$GL_TPINDEX, #1, R0          ; 1312
              0000G  CF            50  C8 000BA        BISL2   R0, LIB$GL_CRE8FLAGS            ;
                                 FF4D  31 000BF        BRW     1$                              ; 1281
                     50            57  D0 000C2  10$:  MOVL    STATUS, R0                      ; 1315
                                       04 000C5        RET                                     ; 1316

; Routine Size:  198 bytes,    Routine Base:  $CODE$ + 05C4
```

```
;  858          1317  1 %SBTTL 'call_tparse & value_req & syntaxerr';
;  859          1318  1
;  860          1319  1 ROUTINE call_tparse (str_desc, states, keys )=
;  861          1320  2 BEGIN
;  862          1321  2 MAP
;  863          1322  2     str_desc : REF BBLOCK;
;  864          1323  2 LOCAL
;  865          1324  2     status;
;  866          1325  2
;  867          1326  2 CH$FILL(0,tpa$k_length0,tpa_block);
;  868          1327  2 tpa_block [tpa$l_count] = tpa$k_count0;
;  869          1328  2 tpa_block [tpa$l_options] = tpa$m_abbrev;
;  870          1329  2 tpa_block [tpa$l_stringcnt] = .str_desc [dsc$w_length];
;  871          1330  2 tpa_block [tpa$l_stringptr] = .str_desc [dsc$a_pointer];
;  872          1331  2 tpa_desc = .str_desc;
;  873          1332  2
;  874          1333  2 status = lib$tparse (tpa_block, .states, .keys);
;  875          1334  2 RETURN .status;
;  876          1335  1 END;                                    !OF call_tparse


                                    007C 00000 CALL_TPARSE:
                                                .WORD   Save R2,R3,R4,R5,R6                     ; 1319
                          56    0000'  CF  9E 00002         MOVAB   TPA_BLOCK, R6
        24          00    6E           00  2C 00007         MOVC5   #0, (SP), #0, #36, TPA_BLOCK    ; 1326
                                         66     0000C
                          66           08  D0 0000D         MOVL    #8, TPA_BLOCK                   ; 1327
                    04    A6           02  D0 00010         MOVL    #2, TPA_BLOCK+4                 ; 1328
                          50     04    AC  D0 00014         MOVL    STR_DESC, R0                    ; 1329
                    08    A6           60  3C 00018         MOVZWL  (R0), TPA_BLOCK+8
                    0C    A6     04    A0  D0 0001C         MOVL    4(R0), TPA_BLOCK+12             ; 1330
                    24    A6           50  D0 00021         MOVL    R0, TPA_DESC                    ; 1331
                          7E     08    AC  7D 00025         MOVQ    STATES, -(SP)                   ; 1333
                                       56  DD 00029         PUSHL   R6
                00000000G   00         03  FB 0002B         CALLS   #3, LIB$TPARSE
                                         04 00032         RET                                     ; 1335

; Routine Size:  51 bytes,    Routine Base:  $CODE$ + 068A

;  877          1336  1
;  878          1337  1 ROUTINE value_req =
;  879          1338  2 BEGIN
;  880          1339  2 lib$gl_valreq = .tpa_block [tpa$l_param];
;  881          1340  2 RETURN true;
;  882          1341  1 END;                                    ! of value_req


                                    0000 00000 VALUE_REQ:
                                                .WORD   Save nothing                            ; 1337
                    0000'  CF     0000'  CF  D0 00002         MOVL    TPA_BLOCK+32, LIB$GL_VALREQ ; 1339
```

```
                             50         01  D0 00009         MOVL    #1, R0                                       ; 1340
                                            04 0000C         RET                                                  ; 1341

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 06BD


;  883          1342  1
;  884          1343  1 ROUTINE syntaxerr =
;  885          1344  2 BEGIN
;  886          1345  2 !
;  887          1346  2 !    This routine is called when TPARSE encounters an error
;  888          1347  2 !
;  889          1348  2 SIGNAL_STOP (.tpa_block [tpa$l_param], 1, .tpa_desc);
;  890          1349  2 RETURN true;
;  891          1350  1 END;



                                          0000 00000 SYNTAXERR:
                                                             .WORD   Save nothing                                 ; 1343
                                 0000'  CF  DD 00002         PUSHL   TPA_DESC                                     ; 1348
                                        01  DD 00006         PUSHL   #1                                           ;
                                 0000'  CF  DD 00008         PUSHL   TPA_BLOCK+32                                 ;
                    00000000G  00       03  FB 0000C         CALLS   #3, LIB$STOP                                 ;
                               50       01  D0 00013         MOVL    #1, R0                                       ; 1349
                                            04 00016         RET                                                  ; 1350

; Routine Size:  23 bytes,    Routine Base:  $CODE$ + 06CA
```

```
1 %SBTTL 'selectcrosref';
1
1 ROUTINE selectcrosref  =
2 BEGIN
2 !
2 ! This routine processes the /CROSS qualifier.
2 !
2 LOCAL
2     firstvalue,
2     optionindex;
2 BIND
2     crefbits = lib$gl_creflags : BITVECTOR;
2
2 firstvalue = true;
2 !
2 ! There were some values, so get and check them
2 !
2 WHILE cli$get_value(sd_cross_reference, token_desc) DO
3     BEGIN
3     IF NOT lib$lookup_key (token_desc, crf_table, !And look it up
3                             optionindex)
3     THEN
4         BEGIN
4         IF NOT lib$lookup_key (token_desc, opt_table, !See if all or none
4                             optionindex)
4         THEN SIGNAL_STOP (                                    !and error if none of these
4                     lib$_badkey,1, token_desc [dsc$w_length])
4         ELSE IF .optionindex EQL 0                            !If /CROSS=ALL
4               THEN lib$gl_creflags = -1                       !then set all flags
4               ELSE lib$gl_creflags = 0;
4          END
3      ELSE crefbits [.optionindex] = true;                     !Set the flag bit
3     firstvalue = false;
2     END;
2 IF .firstvalue
2 THEN
3     BEGIN
3     lib$gl_creflags [lib$v_crfbysym] = true;          !/cross only, so give cref by symbol
3     lib$gl_creflags [lib$v_crfbyval] = true;          ! and cref by value
3     RETURN true;
2     END;
2 RETURN true
1 END;                                                  !of selectcrosref


                                   CREFBITS=          LIB$GL_CREFLAGS


                            003C 00000 SELECTCROSREF:
                                             .WORD    Save R2,R3,R4,R5                    ; 1353
             55       0000'   CF   9E 00002         MOVAB    LIB$GL_CREFLAGS, R5                 ;
             54 00000000G     00   9E 00007         MOVAB    LIB$LOOKUP_KEY, R4                  ;
             53       0000'   CF   9E 0000E         MOVAB    TOKEN_DESC, R3                      ;
             5E               04   C2 00013         SUBL2    #4, SP                              ;
             52               01   D0 00016         MOVL     #1, FIRSTVALUE                      ; 1364
                              53   DD 00019 1$:     PUSHL    R3                                  ; 1368
```

```
                                 0000'  CF  9F 0001B          PUSHAB  SD_CROSS_REFERENCE                    ;
                          0000G  CF        02  FB 0001F          CALLS   #2, CLI$GET_VALUE                    ;
                                 42        50  E9 00024          BLBC    R0, 6$                               ;
                                           5E  DD 00027          PUSHL   SP                                   ; 1370
                                     94    A3  9F 00029          PUSHAB  CRF_TABLE                            ;
                                           53  DD 0002C          PUSHL   R3                                   ;
                                 64        03  FB 0002E          CALLS   #3, LIB$LOOKUP_KEY                   ;
                                 2D        50  E8 00031          BLBS    R0, 4$                               ;
                                           5E  DD 00034          PUSHL   SP                                   ; 1374
                                     B0    A3  9F 00036          PUSHAB  OPT_TABLE                            ;
                                           53  DD 00039          PUSHL   R3                                   ;
                                 64        03  FB 0003B          CALLS   #3, LIB$LOOKUP_KEY                   ;
                                 13        50  E8 0003E          BLBS    R0, 2$                               ;
                                           53  DD 00041          PUSHL   R3                                   ; 1377
                                           01  DD 00043          PUSHL   #1                                   ;
                                0086110C   8F  DD 00045          PUSHL   #8786188                             ;
                    00000000G   00        03  FB 0004B          CALLS   #3, LIB$STOP                         ;
                                           11  11 00052          BRB     5$                                   ;
                                           6E  D5 00054  2$:     TSTL    OPTIONINDEX                          ; 1378
                                           05  12 00056          BNEQ    3$                                   ;
                                 65        01  CE 00058          MNEGL   #1, LIB$GL_CREFLAGS                  ; 1379
                                           08  11 0005B          BRB     5$                                   ;
                                           65  D4 0005D  3$:     CLRL    LIB$GL_CREFLAGS                      ; 1380
                                           04  11 0005F          BRB     5$                                   ; 1370
                          00     65        6E  E2 00061  4$:     BBSS    OPTIONINDEX, CREFBITS, 5$            ; 1382
                                           52  D4 00065  5$:     CLRL    FIRSTVALUE                           ; 1383
                                           B0  11 00067          BRB     1$                                   ; 1368
                                 03        52  E9 00069  6$:     BLBC    FIRSTVALUE, 7$                       ; 1385
                                 65        03  88 0006C          BISB2   #3, LIB$GL_CREFLAGS                  ; 1389
                                 50        01  D0 0006F  7$:     MOVL    #1, R0                               ; 1392
                                               04 00072          RET                                          ; 1393

; Routine Size:  115 bytes,    Routine Base:  $CODE$ + 06E1
```

```
1 %SBTTL 'get_name_list';
1 ROUTINE get_name_list (list_head, qualifier_desc) =
2 BEGIN
2 !
2 ! This routine recalls CLI to get the names of all the keys associated with
2 ! a qualifier (i.e. /DELETE=module1:module2:...:moduleN).  If there are no
2 ! keys, the image is stopped with an error.
2 !
2 ! If the keys are there, they are extracted and put on a list.
2 !
2 MAP
2         list_head : REF VECTOR [2];
2 BUILTIN
2         INSQUE,
2         NULLPARAMETER;
2 LOCAL
2         firstvalue;
2 firstvalue = true;
2 WHILE cli$get_value(.qualifier_desc, token_desc)  DO  !While there are more values
3     BEGIN
3     LOCAL
3         status,
3         namblk : REF BBLOCK;
4     IF NOT (status = lib_get_mem (lnb$c_fixedsize + .token_desc [dsc$w_length],
4                         namblk))        !Allocate memory
3     THEN SIGNAL_STOP (.status);
3     INSQUE (.namblk, .list_head [1]);             !Insert in queue
3     namblk [lnb$b_namlng] = .token_desc [dsc$w_length]; !Set size of name
3     CH$MOVE (.namblk [lnb$b_namlng], .token_desc [dsc$a_pointer],
3                       namblk [lnb$t_name]);                !Copy the name
3     firstvalue = false;
2     END;
2 IF .firstvalue
2 THEN
3     BEGIN
3     IF .NULLPARAMETER(2)
3     THEN SIGNAL_STOP ( lib$_novalue )
3     ELSE SIGNAL_STOP ( lib$_novalue, 1, .qualifier_desc );
2     END;
2 RETURN true
1 END;
```

```
                              01FC 00000 GET_NAME_LIST:
                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8          ; 1395
          58      0000'  CF   9E 00002          MOVAB   TOKEN_DESC, R8                     ;
          57  00000000G  00   9E 00007          MOVAB   LIB$STOP, R7                       ;
          5E             04   C2 0000E          SUBL2   #4, SP                             ;
          56             01   D0 00011          MOVL    #1, FIRSTVALUE                     ; 1411
                         58   DD 00014 1$:      PUSHL   R8                                 ; 1412
                  08     AC   DD 00016          PUSHL   QUALIFIER_DESC                     ;
   0000G  CF             02   FB 00019          CALLS   #2, CLI$GET_VALUE                  ;
          33             50   E9 0001E          BLBC    R0, 3$                             ;
                         5E   DD 00021          PUSHL   SP                                 ; 1417
```

```
                        7E            68  3C 00023          MOVZWL  TOKEN_DESC, -(SP)                 ;
                        6E            0A  C0 00026          ADDL2   #10, (SP)                         ;
                 0000G  CF            02  FB 00029          CALLS   #2, LIB_GET_MEM                   ;
                        05            50  E8 0002E          BLBS    STATUS, 2$                        ;
                                      50  DD 00031          PUSHL   STATUS                            ; 1419
                        67            01  FB 00033          CALLS   #1, LIB$STOP                      ;
                        50      04    AC  D0 00036 2$:      MOVL    LIST_HEAD, R0                     ; 1420
                   04   B0      00    BE  0E 0003A          INSQUE  @NAMBLK, @4(R0)                   ;
                        50            6E  D0 0003F          MOVL    NAMBLK, R0                        ; 1421
                   09   A0            68  90 00042          MOVB    TOKEN_DESC, 9(R0)                 ;
                        51      09    A0  9A 00046          MOVZBL  9(R0), R1                         ; 1422
        0A    A0   04   B8            51  28 0004A          MOVC3   R1, @TOKEN_DESC+4, 10(R0)         ; 1423
                                      56  D4 00050          CLRL    FIRSTVALUE                        ; 1424
                                      C0  11 00052          BRB     1$                                ; 1412
                        2F            56  E9 00054 3$:      BLBC    FIRSTVALUE, 7$                    ; 1426
                        02            6C  91 00057          CMPB    (AP), #2                          ; 1429
                                      05  1E 0005A          BGEQU   4$                                ;
                        50            01  D0 0005C          MOVL    #1, R0                            ;
                                      09  11 0005F          BRB     5$                                ;
                                      50  D4 00061 4$:      CLRL    R0                                ;
                                08    AC  D5 00063          TSTL    8(AP)                             ;
                                      02  12 00066          BNEQ    5$                                ;
                                      50  D6 00068          INCL    R0                                ;
                        08            60  E9 0006A 5$:      BLBC    (R0), 6$                          ;
                          00861104    8F  DD 0006D          PUSHL   #8786180                          ; 1430
                        67            01  FB 00073          CALLS   #1, LIB$STOP                      ;
                                      0E  11 00076          BRB     7$                                ;
                                08    AC  DD 00078 6$:      PUSHL   QUALIFIER_DESC                    ; 1431
                                      01  DD 0007B          PUSHL   #1                                ;
                          00861104    8F  DD 0007D          PUSHL   #8786180                          ;
                        67            03  FB 00083          CALLS   #3, LIB$STOP                      ;
                        50            01  D0 00086 7$:      MOVL    #1, R0                            ; 1433
                                          04 00089          RET                                       ; 1434

; Routine Size:  138 bytes,    Routine Base:  $CODE$ + 0754
```

```
;  979          1435  1 %SBTTL 'clearsqueeze & setsqueeze';
;  980          1436  1 ROUTINE clearsqueeze =
;  981          1437  2 BEGIN
;  982          1438  2 !
;  983          1439  2 ! Clear the local squeeze bit
;  984          1440  2 !
;  985          1441  2 squeeze_flag = 0;
;  986          1442  2 RETURN true
;  987          1443  1 END;


                                     0000 00000 CLEARSQUEEZE:
                                                .WORD   Save nothing                    ; 1436
                                0000'  CF  D4 00002     CLRL    SQUEEZE_FLAG            ; 1441
                                    50     01  D0 00006     MOVL    #1, R0              ; 1442
                                           04  00009     RET                             ; 1443

; Routine Size:  10 bytes,    Routine Base:  $CODE$ + 07DE

;  988          1444  1 ROUTINE setsqueeze =
;  989          1445  2 BEGIN
;  990          1446  2 !
;  991          1447  2 ! Set local squeeze bit
;  992          1448  2 !
;  993          1449  2 squeeze_flag = 1;
;  994          1450  2 RETURN true
;  995          1451  1 END;


                                     0000 00000 SETSQUEEZE:
                                                .WORD   Save nothing                    ; 1444
                                0000'  CF          01  D0 00002     MOVL    #1, SQUEEZE_FLAG        ; 1449
                                    50             01  D0 00007     MOVL    #1, R0                  ; 1450
                                                   04  0000A     RET                             ; 1451

; Routine Size:  11 bytes,    Routine Base:  $CODE$ + 07E8
```

```
;  997          1452  1 %SBTTL 'before_date';
;  998          1453  1 ROUTINE before_date =
;  999          1454  1 !
; 1000          1455  1 ! This routine passes the address of lib$before_date
; 1001          1456  1 !       to date_parse to be set
; 1002          1457  1 !
; 1003          1458  2 BEGIN
; 1004          1459  2 cli$get_value(sd_before, token_desc);
; 1005          1460  2 date_parse( lib$before_date );
; 1006          1461  2 RETURN true;
; 1007          1462  1 END;


                                        0000 00000 BEFORE_DATE:
                                                   .WORD   Save nothing                    ; 1453
                                0000'   CF  9F 00002       PUSHAB  TOKEN_DESC              ; 1459
                                0000'   CF  9F 00006       PUSHAB  SD_BEFORE               ;
                          0000G CF          02 FB 0000A    CALLS   #2, CLI$GET_VALUE       ;
                                0000G   CF  9F 0000F       PUSHAB  LIB$BEFORE_DATE         ; 1460
                          0000V CF          01 FB 00013    CALLS   #1, DATE_PARSE          ;
                                50          01 D0 00018    MOVL    #1, R0                  ; 1461
                                            04 0001B       RET                             ; 1462

; Routine Size:  28 bytes,    Routine Base:  $CODE$ + 07F3

; 1008          1463  1 %SBTTL 'since_date';
```

```
;1010          1464  1 ROUTINE since_date =
;1011          1465  1 !
;1012          1466  1 ! This routine passes the address of lib$since_date
;1013          1467  1 !       to date_parse to be set
;1014          1468  1 !
;1015          1469  2 BEGIN
;1016          1470  2 cli$get_value(sd_since, token_desc);
;1017          1471  2 date_parse( lib$since_date );
;1018          1472  2 RETURN true;
;1019          1473  1 END;


                                        0000 00000 SINCE_DATE:
                                                   .WORD   Save nothing                    ;1464
                                0000'   CF  9F 00002       PUSHAB  TOKEN_DESC              ;1470
                                0000'   CF  9F 00006       PUSHAB  SD_SINCE
                          0000G CF          02 FB 0000A     CALLS   #2, CLI$GET_VALUE
                                0000G   CF  9F 0000F       PUSHAB  LIB$SINCE_DATE          ;1471
                          0000V CF          01 FB 00013     CALLS   #1, DATE_PARSE
                                50          01 D0 00018     MOVL    #1, R0                  ;1472
                                            04 0001B        RET                             ;1473

; Routine Size:  28 bytes,    Routine Base:  $CODE$ + 080F

;1020          1474  1 %SBTTL 'date_parse';
;1021          1475  1 ROUTINE date_parse  ( date ) =
;1022          1476  1 !----
;1023          1477  1 !
;1024          1478  1 ! Functional description
;1025          1479  1 !
;1026          1480  1 !       This routine parses the value on the /SINCE and /BEFORE
;1027          1481  1 !       qualifiers.  It converts the ASCII value of the qualifier
;1028          1482  1 !       into 64-bit system date format.  The value can be a date
;1029          1483  1 !       string (such as dd-mmm-yyyy hh:mm:ss.s) or the keywords
;1030          1484  1 !       YESTERDAY or TODAY.
;1031          1485  1 !
;1032          1486  1 ! Output parameters
;1033          1487  1 !
;1034          1488  1 !       lib$since_date = Date if /SINCE specified.
;1035          1489  1 !       lib$before_date = Date if /BEFORE specified.
;1036          1490  1 !
;1037          1491  1 !----
;1038          1492  2 BEGIN
;1039          1493  2
;1040          1494  2 BIND
;1041          1495  2     date_field = .date : ref bblock;
;1042          1496  2 LOCAL
;1043          1497  2     status;                               ! Status return value
;1044          1498  2
;1045          1499  2 !
;1046          1500  2 !       Convert the ASCII string into 64-bit system time format.
;1047          1501  2 !
;1048          1502  2 status = lib$cvt_time(token_desc, date_field);
```

```
; 1049          1503  2
; 1050          1504  2 IF NOT .status                          ! If error in conversion,
; 1051          1505  2 THEN
; 1052          1506  2     SIGNAL_STOP(.status);               ! then output error message
; 1053          1507  2
; 1054          1508  2 RETURN true;
; 1055          1509  1 END;


                                        0000 00000 DATE_PARSE:
                                                           .WORD   Save nothing                        ; 1475
                                   04    AC  DD 00002      PUSHL   DATE                                ; 1502
                                 0000'   CF  9F 00005      PUSHAB  TOKEN_DESC                          ;
                    00000000G  00        02  FB 00009      CALLS   #2, LIB$CVT_TIME                    ;
                               09        50  E8 00010      BLBS    STATUS, 1$                          ; 1504
                                         50  DD 00013      PUSHL   STATUS                              ; 1506
                    00000000G  00        01  FB 00015      CALLS   #1, LIB$STOP                        ;
                               50        01  D0 0001C 1$:  MOVL    #1, R0                              ; 1508
                                             04 0001F      RET                                         ; 1509

; Routine Size:  32 bytes,    Routine Base:  $CODE$ + 082B
```

```
1 %SBTTL 'allocate_fbd & initialize_fdb';
1 ROUTINE allocate_fdb (fdbadr) =
2 BEGIN
2 !
2 ! This routine allocates a file descriptor block and initializes it.
2 !
2 LOCAL
2       status;
2
3 IF NOT (status = lib_get_mem (fdbsize, .fdbadr))        !Allocate the FDB
2 THEN SIGNAL_STOP (.status);
2 initialize_fdb (..fdbadr);                              !and initialize it
2 RETURN true
1 END;


                                   0000 00000 ALLOCATE_FDB:
                                                      .WORD    Save nothing                     ; 1511
                       04   AC  DD 00002              PUSHL    FDBADR                           ; 1519
          7E    029E   8F  3C 00005                   MOVZWL   #670, -(SP)                      ;
   0000G  CF               02  FB 0000A               CALLS    #2, LIB_GET_MEM                  ;
          09               50  E8 0000F               BLBS     STATUS, 1$                       ;
                           50  DD 00012               PUSHL    STATUS                           ; 1520
000000006 00               01  FB 00014               CALLS    #1, LIB$STOP                     ;
                       04  BC  DD 0001B 1$:           PUSHL    @FDBADR                          ; 1521
   0000V  CF               01  FB 0001E               CALLS    #1, INITIALIZE_FDB               ;
          50               01  D0 00023               MOVL     #1, R0                           ; 1522
                               04 00026               RET                                       ; 1523

; Routine Size:  39 bytes,    Routine Base:  $CODE$ + 084B


1 ROUTINE initialize_fdb (fdbadr) =
2 BEGIN
2 !
2 ! This routine initializes a pre-allocated FDB
2 !
2 MAP
2     fdbadr : REF BBLOCK;
2 LOCAL
2       namblk : REF BBLOCK,                              !Pointer to NAM block
2       rsblk  : REF BBLOCK;                              !Pointer to resultant name string area
2 namblk = fdbadr [fdb$t_nam];                            !Point to NAM block part
2 rsblk = .namblk + nam$c_bln;                            !Point to resultant string area
2 CH$FILL (0, fdbsize, .fdbadr);                          !Zero the FDB
2 namblk [nam$l_rsa] = .rsblk;                            !Store RSA address
2 namblk [nam$b_rss] = nam$c_maxrss;                      !And its length
2 namblk [nam$l_esa] = .rsblk+nam$c_maxrss;               !Expanded name string goes
2 namblk [nam$b_ess] = nam$c_maxrss;                      ! in same place so errors
2                                                         ! messages are correct.
2 namblk [nam$b_bid] = nam$c_bid;                         !Identify it as a NAM
2 namblk [nam$b_bln] = nam$c_bln;                         !...
2 RETURN true
1 END;
```

```
                                                    00FC 00000 INITIALIZE_FDB:
                                                                       .WORD   Save R2,R3,R4,R5,R6,R7               ; 1524
                  56        04    AC 00000040   8F    C1 00002         ADDL3   #64, FDBADR, NAMBLK                  ; 1534
                                  57       60   A6    9E 0000B         MOVAB   96(R6), RSBLK                        ; 1535
   029E    8F     00              6E            00    2C 0000F         MOVC5   #0, (SP), #0, #670, @FDBADR          ; 1536
                                           04   BC       00016                                                      ;
                            04    A6            57    D0 00018         MOVL    RSBLK, 4(NAMBLK)                     ; 1537
                            0C    A6     00FF   C7    9E 0001C         MOVAB   255(R7), 12(NAMBLK)                  ; 1539
                            0A    A6            01    8E 00022         MNEGB   #1, 10(NAMBLK)                       ; 1540
                                  66            02    90 00026         MOVB    #2, (NAMBLK)                         ; 1542
                            01    A6     FF60   8F    B0 00029         MOVW    #65376, 1(NAMBLK)                    ; 1543
                                  50            01    D0 0002F         MOVL    #1, R0                               ; 1544
                                                      04 00032         RET                                          ; 1545

; Routine Size:  51 bytes,    Routine Base:  $CODE$ + 0872
```

```
1 %SBTTL 'set_*_type';
1 ROUTINE set_lib_type (type, defext) =
2 BEGIN
2 !
2 ! This routine sets the default library type and default extension
2 !
2 MAP
2     defext : REF VECTOR [4];
2 lib$gl_type = .type;                    !Set library type
2 CH$MOVE (dsc$c_s_bln, defext [0], def_lib_extn);
2 CH$MOVE (dsc$c_s_bln, defext [2], def_fil_extn);
2 RETURN true
1 END;
```

```
                                        007C 00000 SET_LIB_TYPE:
                                                   .WORD   Save R2,R3,R4,R5,R6          ; 1547
        0000G  CF        04  AC  D0 00002          MOVL    TYPE, LIB$GL_TYPE            ; 1554
               56        08  AC  D0 00008          MOVL    DEFEXT, R6                   ; 1555
  0000'  CF    66            08  28 0000C          MOVC3   #8, (R6), DEF_LIB_EXTN       ;
  0000'  CF    08  A6        08  28 00012          MOVC3   #8, 8(R6), DEF_FIL_EXTN      ; 1556
               50            01  D0 00019          MOVL    #1, R0                       ; 1557
                                 04 0001C          RET                                  ; 1558

; Routine Size:  29 bytes,    Routine Base:  $CODE$ + 08A5
```

```
1 ROUTINE set_text_type =
1 !
1 ! This routine is called when the /text qualifier is processed.
1 !
1 RETURN set_lib_type (lib$s_text, lib_txt_defext);
```

```
                                        0000 00000 SET_TEXT_TYPE:
                                                   .WORD   Save nothing                 ; 1559
                      0000G  CF  9F 00002          PUSHAB  LIB_TXT_DEFEXT               ; 1563
                             04  DD 00006          PUSHL   #4                           ;
               D7  AF        02  FB 00008          CALLS   #2, SET_LIB_TYPE             ;
                                 04 0000C          RET                                  ;

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 08C2
```

```
1 ROUTINE set_help_type =
1 !
1 ! This routine is called when the /HELP qualifier is processed.
1 !
1 RETURN set_lib_type (lib$s_help, lib_hlp_defext);
```

```
                                        0000 00000 SET_HELP_TYPE:
                                                   .WORD   Save nothing                          ; 1564
                                0000G   CF  9F 00002       PUSHAB  LIB_HLP_DEFEXT               ; 1568
                                        03  DD 00006       PUSHL   #3                           ;
                             CA   AF            02  FB 00008       CALLS   #2, SET_LIB_TYPE     ;
                                                04  0000C       RET                             ;

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 08CF

; 1117          1569  1 ROUTINE set_macro_type =
; 1118          1570  1 !
; 1119          1571  1 ! This routine is called when the /MACRO qualifier is processed.
; 1120          1572  1 !
; 1121          1573  1 RETURN set_lib_type (lib$s_macro, lib_mac_defext);



                                        0000 00000 SET_MACRO_TYPE:
                                                   .WORD   Save nothing                          ; 1569
                                0000G   CF  9F 00002       PUSHAB  LIB_MAC_DEFEXT               ; 1573
                                        02  DD 00006       PUSHL   #2                           ;
                             BD   AF            02  FB 00008       CALLS   #2, SET_LIB_TYPE     ;
                                                04  0000C       RET                             ;

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 08DC

; 1122          1574  1 ROUTINE set_object_type =
; 1123          1575  1 !
; 1124          1576  1 ! This routine is called when the /OBJECT qualifier is processed.
; 1125          1577  1 !
; 1126          1578  1 RETURN set_lib_type (lib$s_object, lib_obj_defext);



                                        0000 00000 SET_OBJECT_TYPE:
                                                   .WORD   Save nothing                          ; 1574
                                0000G   CF  9F 00002       PUSHAB  LIB_OBJ_DEFEXT               ; 1578
                                        01  DD 00006       PUSHL   #1                           ;
                             B0   AF            02  FB 00008       CALLS   #2, SET_LIB_TYPE     ;
                                                04  0000C       RET                             ;

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 08E9

; 1127          1579  1 ROUTINE set_shr_type  =
; 1128          1580  1 !
; 1129          1581  1 ! This routine is called when the /SHARE qualifier is processed.
; 1130          1582  1 !
```

```
; 1131          1583  1 RETURN set_lib_type (lib$s_shrstb, lib_shr_defext);

                                    0000 00000 SET_SHR_TYPE:
                                               .WORD   Save nothing                    ; 1579
                                0000G  CF  9F 00002    PUSHAB  LIB_SHR_DEFEXT          ; 1583
                                       05  DD 00006    PUSHL   #5                      ;
                              A3   AF        02  FB 00008    CALLS   #2, SET_LIB_TYPE   ;
                                             04 0000C    RET                             ;

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 08F6

; 1132          1584  1
```

```
; 1134          1585  1 ROUTINE clierror ( errorcode ) : novalue =
; 1135          1586  2 BEGIN
; 1136          1587  2 !
; 1137          1588  2 !       This routine is called by the result parser if it detects any
; 1138          1589  2 !       error in the command.  REQUESTDESC is the address of the current
; 1139          1590  2 !       parameter descriptor and ERRORCODE is the encoded reason for
; 1140          1591  2 !       the error.
; 1141          1592  2 !
; 1142          1593  2 SIGNAL_STOP (.errorcode);                           !Print error message and quit
; 1143          1594  2 RETURN;
; 1144          1595  1 END;


                                        0000 00000 CLIERROR:
                                                          .WORD   Save nothing                          ; 1585
                                  04   AC  DD 00002       PUSHL   ERRORCODE                             ; 1593
                  00000000G  00        01  FB 00005       CALLS   #1, LIB$STOP                          ;
                                           04 0000C       RET                                           ; 1595

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 0903


; 1145          1596  1 END                                 ! Of module
; 1146          1597  0 ELUDOM


                                                          .EXTRN  LIB$SIGNAL, LIB$STOP

;                              PSECT SUMMARY
;
;       Name                 Bytes                      Attributes
;
;  $PLIT$                      540  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $GLOBAL$                     12  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                      1116  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  _LIB$KEY0$                   32  NOVEC,NOWRT,  RD , EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  _LIB$STATE$                 371  NOVEC,NOWRT,  RD , EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  _LIB$KEY1$                  123  NOVEC,NOWRT,  RD , EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  $CODE$                     2320  NOVEC,NOWRT,  RD , EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                         Library Statistics
;
;                                        -------- Symbols --------      Pages       Processing
;       File                             Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1      9776      103         1        581        00:01.0
;  _$255$DUA28:[SYSLIB]CLIMAC.L32;1         14        0         0          9        00:00.1
;  _$255$DUA28:[SYSLIB]TPAMAC.L32;1         42       25        59         14        00:00.1
```

```
;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:GETCMD/OBJ=OBJ$:GETCMD MSRC$:GETCMD/UPDATE=(ENH$:GETCMD)

; Size:         2320 code + 2194 data bytes
; Run Time:        01:07.7
; Elapsed Time:    02:18.8
; Lines/CPU Min:    1415
; Lexemes/CPU-Min: 38989
; Memory Used:  362 pages
; Compilation Complete
```

GETCMD
LIS

INPUTOBJ
LIS

INPUTTXT
LIS

LIBRARIAN
LIS

INPUTMAC
LIS

INPUTHLP
LIS